简单线程池的实现

作者 小鄧子 (/u/df40282480b4) + 关注 2015.03.14 23:23* 字数 697 阅读 2050 评论 6 喜欢 48

(/u/df40282480b4)

本篇主要用作链接使用。

线程池的核心就是,当接收到一个任务,需要一个线程去执行时,并不着急创建 一个线程,而是先去线程池检查是否有空闲的线程,如果有,则直接使用,如果 没有则创建新的线程,然后执行任务。任务执行完成后,也不着急销毁线程,而 是将线程放到线程池的空闲队列中,等待下次使用。

首先是线程池的实现。

首先是线程池的实现:

```
* Created by Joker on 2015/3/7.
public class ThreadPool {
 private static ThreadPool instance = null;
 //空闲的线程队列
private List<JThread> idleThreads;
//已有的线程总数
private int threadCounter = 0;
 private Boolean isShunDown = false;
 public ThreadPool() {
    //初始化空闲线程队列容量为5
    this.idleThreads = new Vector<JThread>(5);
    this.threadCounter = 0;
private static class SingleTonHolder {
    private static ThreadPool threadPool = new ThreadPool();
 /*单例获得线程池实例*/
 public static ThreadPool getInstance() {
    return SingleTonHolder.threadPool;
public int getThreadCounter() {
    return threadCounter;
 * 将线程放入池中, 回收线程
 protected \ synchronized \ void \ repool(JThread \ repoolingThread) \ \{
    if (!isShunDown) {
        idleThreads.add(repoolingThread);
    } else {
        repoolingThread.shutDown();
}
 * 停止池中所有线程
public synchronized void shutDownAll() {
    this.isShunDown = true;
    for (JThread jThread : idleThreads) {
        jThread.shutDown();
}
 * 执行线程任务
 public synchronized void execute(Runnable target) {
   this.isShunDown = false;
    JThread jThread = null;
    /*如果池中有空余线程,直接使用该线程*/
    if (idleThreads.size() > 0) {
        jThread = idleThreads.get(idleThreads.size() - 1);
        //将该线程从池中移除
        idleThreads.remove(jThread);
        //立即执行该任务
        jThread.setTarget(target);
    }
    /*没有空闲线程, 创建新线程*/
    else {
        threadCounter++;
        jThread = new JThread(target, "JThread:" + threadCounter, ThreadPool.this);
        jThread.start();
    }
}
}
```

要使用上述线程池,还需要一个永不退出的工作现场与之配合。是一个 While 循环,手动关闭之前,永不结束,一直等待新的任务进来。

该工作线程的实现如下:

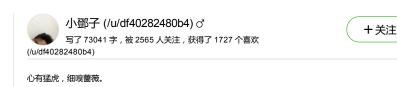
```
* Created by Joker on 2015/3/7.
public class JThread extends Thread {
//线程池
private ThreadPool threadPool;
private Runnable target;
private boolean isShutDown = false;
private boolean isIdle = false;
public JThread(Runnable target, String name, ThreadPool threadPool) {
    super(name);
    this.target = target;
    this.threadPool = threadPool;
public Runnable getTarget() {
    return target;
public boolean isIdle() {
    return isIdle;
@Override
public void run() {
    //只要没有关闭,则一直不结束该线程
    while (!isShutDown) {
       isIdle = false;
       if (null != target) {
           //执行任务,注意这里调用的是run(),而不是start()
            target.run();
        //任务结束,闲置状态
        isIdle = true;
           threadPool.repool(JThread.this);
           synchronized (this) {
               //线程空闲,等待新的任务到来
               wait();
        } catch (InterruptedException e) {
        isIdle = false;
}
public synchronized void setTarget(Runnable target) {
   this.isShutDown = false;
    this.target = target;
    //设置任务之后,通知run方法,开始执行任务
    notifyAll();
}
 * 关闭线程
public synchronized void shutDown() {
    this.isShutDown = true;
   notifyAll();
}
```

使用方法如下:

在线程频繁调度时,通过线程的复用,对性能的提升有很大的帮 助

■ Android原创集 (/nb/575116)

举报文章 © 著作权归作者所有



♡ 喜欢 (/sign_in?utm_source=desktop&utm_medium=not-signed-in-like-button)







更多分享

(http://cwb.assets.jianshu.io/notes/images/790483/we

48

▮被以下专题收入,发现更多相似内容

android (/c/affc7c38a3e8?utm_source=desktop&utm_medium=notes-included-collection)

Android世界 (/c/81ccda347cc6?utm_source=desktop&utm_medium=notes-included-collection)

Android... (/c/2d3730e6ec3b?utm_source=desktop&utm_medium=notes-included-collection)

Android知识 (/c/3fde3b545a35?utm_source=desktop&utm_medium=notes-included-collection)

Android... (/c/680c4a350492?utm_source=desktop&utm_medium=notes-included-collection)

Mndroid开发 (/c/d1591c322c89?utm_source=desktop&utm_medium=notes-included-collection)

ITBox (/c/7c3a1148bd70?utm_source=desktop&utm_medium=notes-included-collection)

展开更多 ~