Java中的线程池

使用线程池的优点

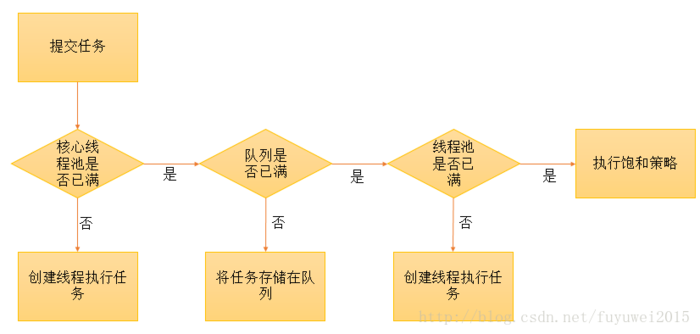
降低资源消耗

提高响应速度

提高线程的可管理性

线程池的处理流程

*/\*  
 \* Proceed in 3 steps:  
 \*  
 \* 1. If fewer than corePoolSize threads are running, try to  
 \* start a new thread with the given command as its first  
 \* task. The call to addWorker atomically checks runState and  
 \* workerCount, and so prevents false alarms that would add  
 \* threads when it shouldn't, by returning false.  
 \*  
 \* 2. If a task can be successfully queued, then we still need  
 \* to double-check whether we should have added a thread  
 \* (because existing ones died since last checking) or that  
 \* the pool shut down since entry into this method. So we  
 \* recheck state and if necessary roll back the enqueuing if  
 \* stopped, or start a new thread if there are none.  
 \*  
 \* 3. If we cannot queue task, then we try to add a new  
 \* thread. If it fails, we know we are shut down or saturated  
 \* and so reject the task.  
 \*/*



源码分析

