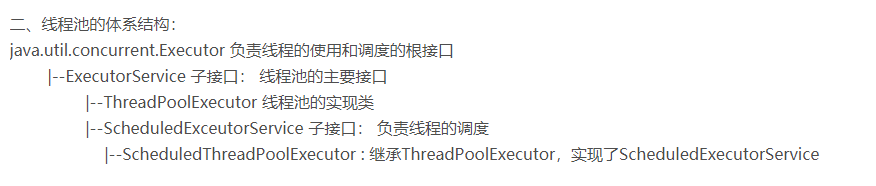
# 线程池

## 线程池的作用：

线程池作用就是限制系统中执行线程的数量。  


## 为什么要用线程池:

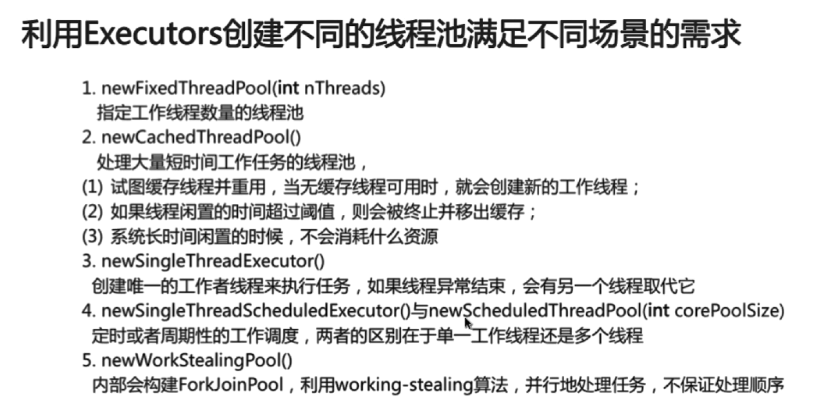
1.减少了创建和销毁线程的次数，每个工作线程都可以被重复利用，可执行多个任务。

2.可以根据系统的承受能力，调整线程池中工作线线程的数目，防止因为消耗过多的内存，而把服务器累趴下(每个线程需要大约1MB内存，线程开的越多，消耗的内存也就越大，最后死机)。

Java里面线程池的顶级接口是Executor，但是严格意义上讲Executor并不是一个线程池，而只是一个执行线程的工具。真正的线程池接口是ExecutorService。

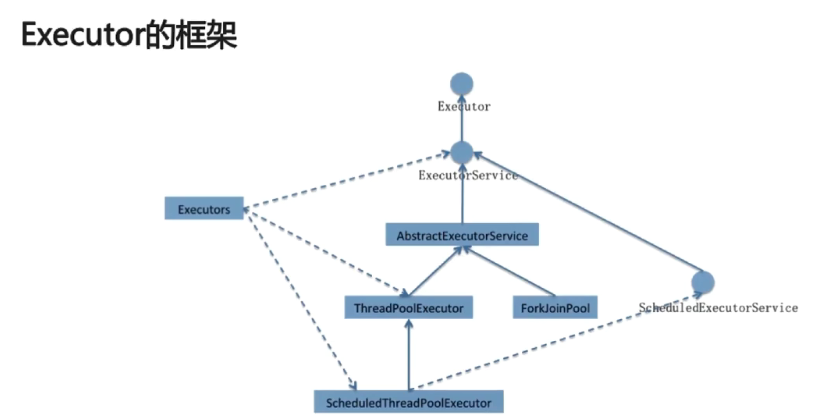
# Executors

## Executors创建不同的线程池



## Executors的框架

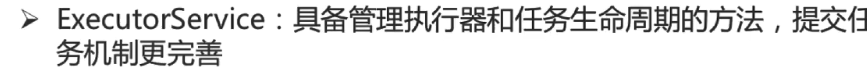
### 图解



### Executor



### ExecutorService

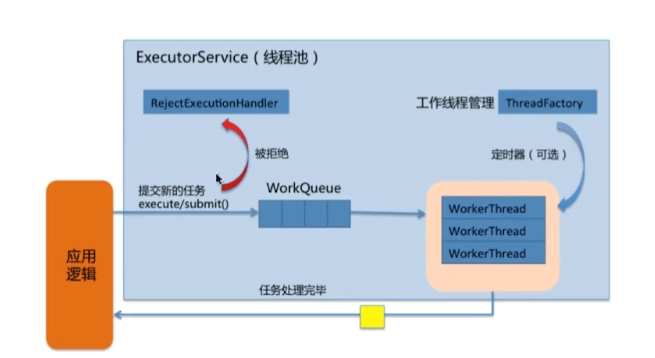


### ScheduledExecutorService

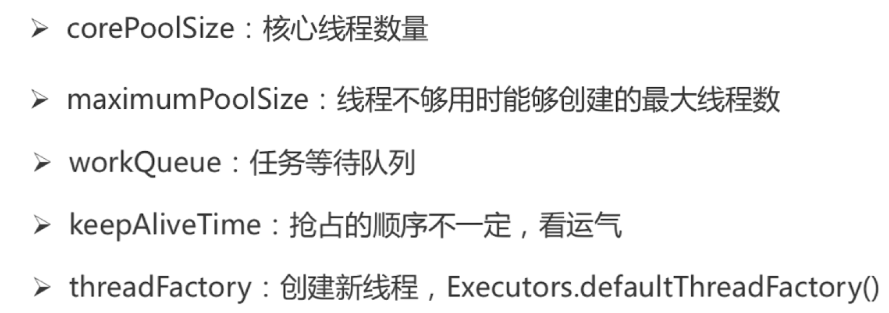


# ThreadPoolExecutor

## 流程



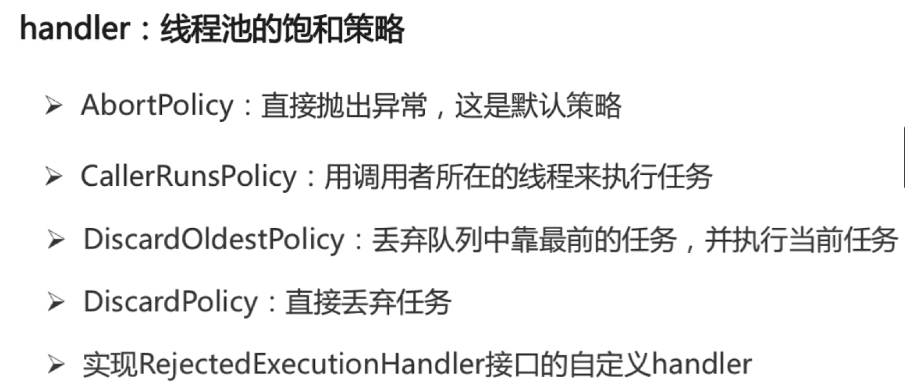
## 构造函数



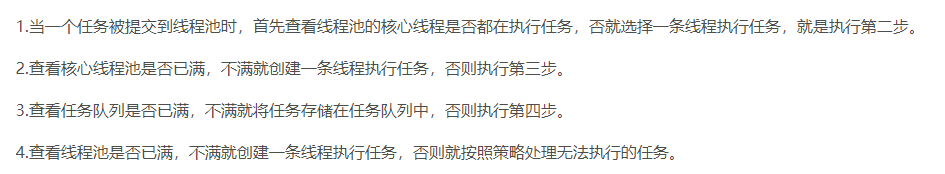
## 队列-最大线程数-核心线程数

<https://blog.csdn.net/sunRest/article/details/103926044>

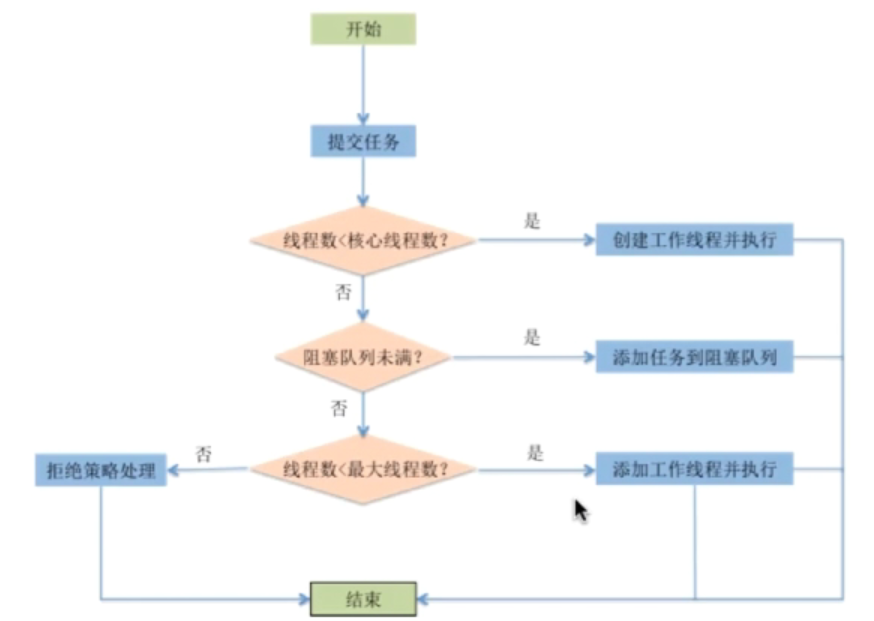
### 拒绝策略（线程池的饱和策略）



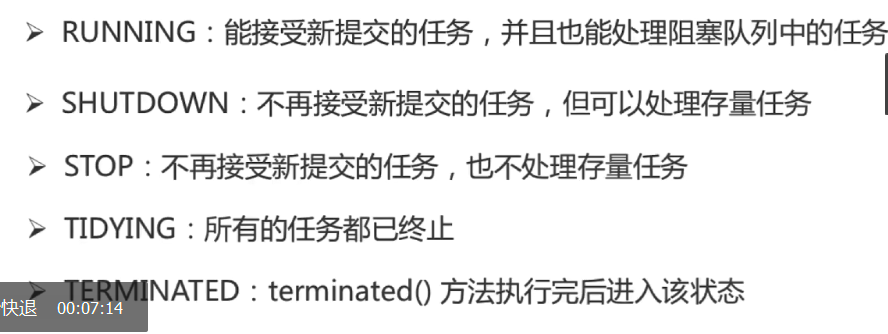
### 处理流程



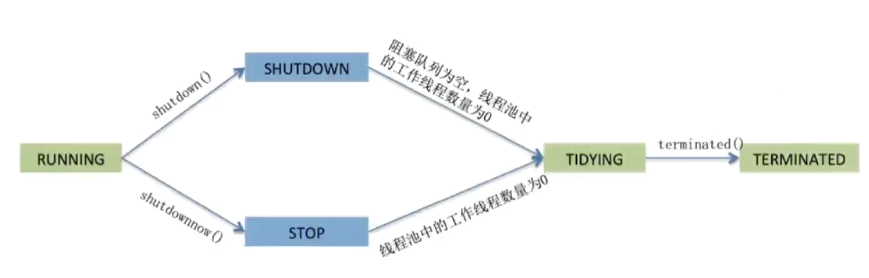
### 处理流程图



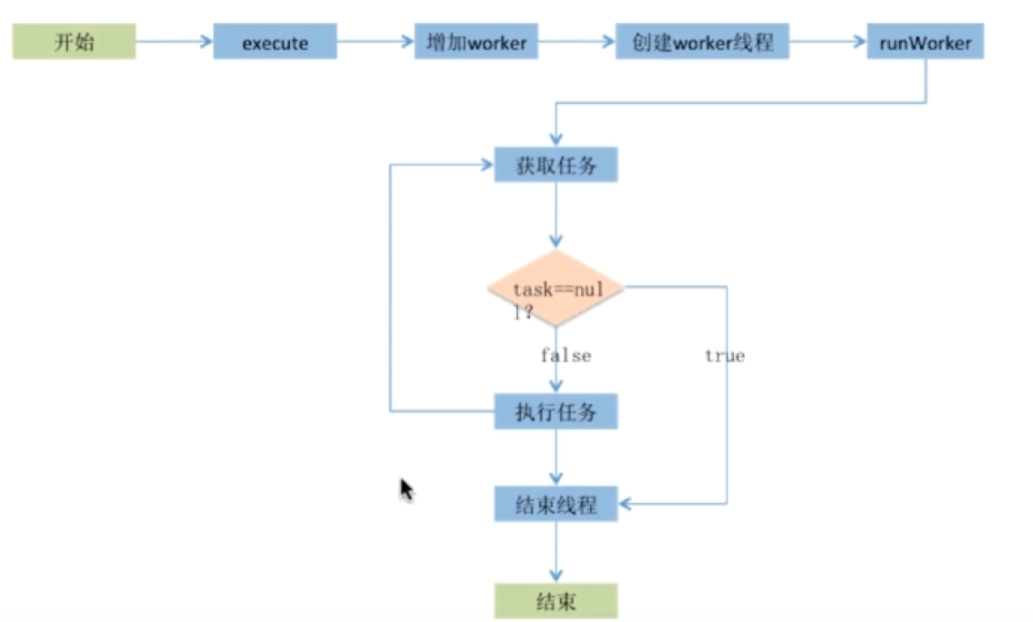
### 线程池的状态



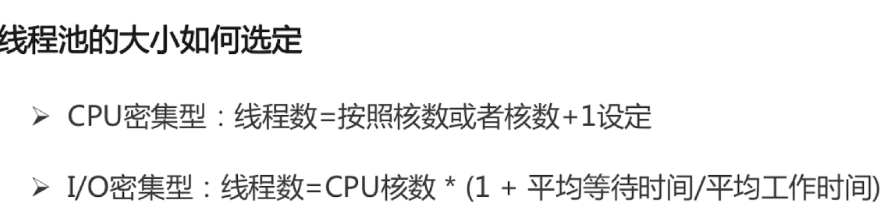
### 状态转换图



### 工作线程的生命周期



## 线程池大小如何选定



# 创建线程池

https://www.cnblogs.com/frankyou/p/9467905.html

## ExecutorService

### 固定线程数的线程池newFixedThreadPool

创建固定大小的线程池。每次提交一个任务就创建一个线程，直到线程达到线程池的最大大小。线程池的大小一旦达到最大值就会保持不变，如果某个线程因为执行异常而结束，那么线程池会补充一个新线程

|  |
| --- |
| **public class** TestFixedThreadPool {  **public static void** main(String[] args) {  *//创建一个可重用固定线程数的线程池* ExecutorService pool = Executors.*newFixedThreadPool*(2);  *//创建实现了Runnable接口对象，Thread对象当然也实现了Runnable接口* Thread t1 = **new** MyThread();  Thread t2 = **new** MyThread();  Thread t3 = **new** MyThread();  Thread t4 = **new** MyThread();  Thread t5 = **new** MyThread();  *//将线程放入池中进行执行* pool.execute(t1);  pool.execute(t2);  pool.execute(t3);  pool.execute(t4);  pool.execute(t5);  *//关闭线程池* pool.shutdown();  } } |

### 单个线程的线程池newSingleThreadExecutor

创建一个单线程的线程池。这个线程池只有一个线程在工作，也就是相当于单线程串行执行所有任务。如果这个唯一的线程因为异常结束，那么会有一个新的线程来替代它。此线程池保证所有任务的执行顺序按照任务的提交顺序执行

|  |
| --- |
| **public class** TestSingleThreadExecutor {  **public static void** main(String[] args) {  *//创建一个可重用固定线程数的线程池* ExecutorService pool = Executors.*newSingleThreadExecutor*();  *//创建实现了Runnable接口对象，Thread对象当然也实现了Runnable接口* Thread t1 = **new** MyThread();  Thread t2 = **new** MyThread();  Thread t3 = **new** MyThread();  Thread t4 = **new** MyThread();  Thread t5 = **new** MyThread();  *//将线程放入池中进行执行* pool.execute(t1);  pool.execute(t2);  pool.execute(t3);  pool.execute(t4);  pool.execute(t5);  *//关闭线程池* pool.shutdown();  } } |

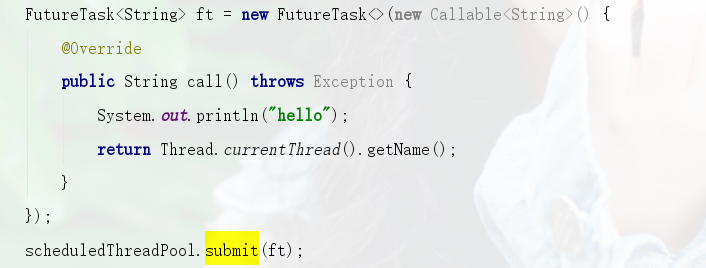
### 缓存的线程池（newCachedThreadPool）

## ScheduledExecutorService

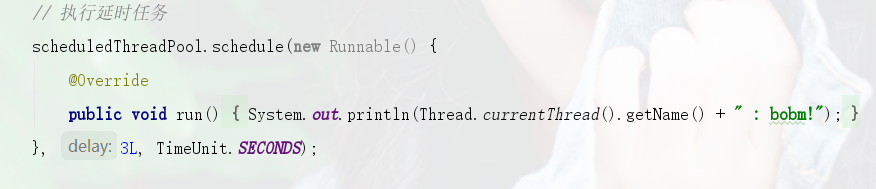
### 固定个数的线程池（newScheduledThreadPool）

相比于第一个固定个数的线程池强大在  **①可以执行延时任务，②也可以执行带有返回值的任务**

#### submit(); 执行带有返回值的任务



#### schedule() 用来执行延时任务.



## 注意

### 手动创建线程池

https://www.jianshu.com/p/0e228dc30793



