#进程

实现多任务的方式：

多线程模式

多进程模式

协程模式

多进程+多线程模式

对于操作系统而言，一个任务就是一个进程

进程是系统中程序执行和资源分配的基本单元，每个进程都有自己的数据段，代码段，堆栈段

进程间数据是独立的，可以借助队列或者管道实现通信

import multiprocessing

m = multiprocessing.Manager()

d = m.dict()

p = multiprocessing.Process(target = camerashow3,args = (1,d))

p.start()

使用：

image = d[1]

# -\*- coding:UTF-8 -\*-  
**from** multiprocessing **import** Process,Queue  
**import** os,time  
  
**def write**(q):  
 print("启动写子进程%s"%(os.getpid()))#获取当前进程id  
 **for** chr **in** ['A','B','C','D']:  
 q.put(chr)  
 time.sleep(1)  
 print("启动写子进程结束%s"%(os.getpid()))  
**def read**(q):  
 print("启动读子进程%s" % (os.getpid())) # 获取当前进程id  
 **while True**:  
 value = q.get(**True**)  
 print('value='+ value)  
 print("结束读子进程%s" % (os.getpid())) # 获取当前进程id  
**if** \_\_name\_\_ == "\_\_main\_\_":  
 q= Queue()  
 pw = Process(target=write,args=(q,))  
 pr = Process(target=read,args=(q,))  
 pw.start()  
 pr.start()  
 pw.join()  
 #pw进程是个死循环，无法等待其结束，只能强行结束  
 pr.terminate()  
 print("父进程结束")

#线程

在一个进程的内部同时干多件事

就需要同时运行多个“子任务”，我们把进程内的这些“子任务”叫做线程

线程通常叫做轻型的进程，线程是共享内存空间的并发执行的多任务，每一个线程都共享一个进程的资源

进程有独立的堆栈

线程是最小的执行单元，而进程由至少一个线程组成，如何调度进程和线程，完全由操作系统决定，程序不能决定什么时候执行，执行多长时间

模块：

1. \_thread模块 低级模块
2. threading模块 高级模块 对\_thread进行了封装