**thread(多现程)**

GIL的全称是Global Interpreter Lock(全局解释器锁)，每个CPU在同一时间只能执行一个线程。

适用于IO密集型代码（文件处理，网络爬虫）：多线程能够有效提升效率(单线程下有IO操作会进行IO等待，造成不必要的时间浪费，而开启多线程能在线程A等待时，自动切换到线程B，可以不浪费CPU的资源，从而能提升程序执行效率)

线程同步：利用锁定和解锁

**import** threading  
**import** time  
  
  
**class** myThread(threading.Thread):  
 **def** \_\_init\_\_(self,threadID,name,counter):  
 threading.Thread.\_\_init\_\_(self)  
 self.threadID = threadID  
 self.name=name  
 self.counter=counter  
 **def** run(self):  
 print(**"Start "**+self.name)  
 **threadLock.acquire()** print\_time(self.name,self.counter,5)  
 print(**"Exit"**)  
 **threadLock.release()**  
**def** print\_time(threadName,delay,counter):  
 **while** counter:  
 **if** 0:  
 threading.exit()  
 time.sleep(delay)  
 print(**"%s: %s"** %(threadName,time.ctime(time.time())))  
 counter-=1  
  
**threadLock=threading.Lock()**  
threads=[]  
  
thread1=myThread(1,**"Thread-1"**,1)  
thread2=myThread(2,**"Thread-2"**,2)  
  
thread1.start()  
thread2.start()  
  
threads.append(thread1)  
threads.append(thread2)  
  
**for** t **in** threads:  
 t.join()

输出结果：

Start Thread-1

Start Thread-2

Thread-1: Wed May 31 20:35:31 2017

Thread-1: Wed May 31 20:35:32 2017

Thread-1: Wed May 31 20:35:33 2017

Thread-1: Wed May 31 20:35:34 2017

Thread-1: Wed May 31 20:35:35 2017

Exit

Thread-2: Wed May 31 20:35:37 2017

Thread-2: Wed May 31 20:35:39 2017

Thread-2: Wed May 31 20:35:41 2017

Thread-2: Wed May 31 20:35:43 2017

Thread-2: Wed May 31 20:35:45 2017

Exit