**1、进程、线程和协程的区别和联系**

|  | **进程** | **线程** | **协程** |
| --- | --- | --- | --- |
| 定义 | 资源分配和拥有的基本单位 | 程序执行的基本单位 | 用户态的轻量级线程，线程内部调度的基本单位 |
| 切换情况 | 进程CPU环境(栈、寄存器、页表和文件句柄等)的保存以及新调度的进程CPU环境的设置 | 保存和设置程序计数器、少量寄存器和栈的内容 | 先将寄存器上下文和栈保存，等切换回来的时候再进行恢复 |
| 切换者 | 操作系统 | 操作系统 | 用户 |
| 切换过程 | 用户态->内核态->用户态 | 用户态->内核态->用户态 | 用户态(没有陷入内核) |
| 调用栈 | 内核栈 | 内核栈 | 用户栈 |
| 拥有资源 | CPU资源、内存资源、文件资源和句柄等 | 程序计数器、寄存器、栈和状态字 | 拥有自己的寄存器上下文和栈 |
| 并发性 | 不同进程之间切换实现并发，各自占有CPU实现并行 | 一个进程内部的多个线程并发执行 | 同一时间只能执行一个协程，而其他协程处于休眠状态，适合对任务进行分时处理 |
| 系统开销 | 切换虚拟地址空间，切换内核栈和硬件上下文，CPU高速缓存失效、页表切换，开销很大 | 切换时只需保存和设置少量寄存器内容，因此开销很小 | 直接操作栈则基本没有内核切换的开销，可以不加锁的访问全局变量，所以上下文的切换非常快 |
| 通信方面 | 进程间通信需要借助操作系统 | 线程间可以直接读写进程数据段(如全局变量)来进行通信 | 共享内存、消息队列 |