

# 实习的收获

---

xbZhong

2025-08-29

[本页PDF](#)

## 线程、进程、协程、事件循环

---

**进程**（资源分配基本单位）

- 进程可以看作是一个程序，比如小红书、王者荣耀这些，**进程有自己的独立堆栈空间**，进程间**内存不共享**，如果需要则需要**进程通信**
- 进程**创立/销毁都需要分配/释放资源**，进程之间具有**隔离性**

**线程**（调度执行基本单位）

- 熟悉CPU可以从CPU核心入手，一个核心可以处理一个线程，**多核心就可以执行多线程任务**
  - 线程**并非越多越好**，因为线程本质上是会**抢占CPU资源执行任务的**，表面上我们看起来是并行执行任务，实际上是**CPU在瞬间不断切换去处理各个线程导致的**
- 线程共享进程资源，这样会导致**资源竞争问题**，多个线程**访问同一资源**，**管理不当会导致线程崩溃**，常见方法有加锁（易导致死锁）
- 一个进程中有多个线程，线程比进程更加轻量，**线程是抢占式的**

**协程**

- **比线程更加轻量，是由程序调度的**，协程是**协作式的**
  - 比如python，使用 `await` 的时候**协程会让出CPU给其它协程执行任务**，等到**I/O（耗时操作，如文件读写等）操作结束后再执行自己的任务**
  - go、cpp都有自己的协程接口，这就是为什么大厂做后端用go比较多
- **多个协程在一个线程内交替执行**，一个线程同一时刻**只有一个协程进行**

**事件循环**

- 在js中较常见
- **先处理同步任务，再处理异步任务**
  - 异步任务会有回调函数，异步任务执行完后会把**回调函数放到任务队列里**
  - 引擎执行完同步任务就会去**任务队列取回调函数，执行异步任务**
- 微任务和宏任务都是**异步任务**
  - 微任务等级**高**，宏任务等级**低**
  - 引擎做完同步任务会执行完**微任务队列的全部回调函数**
  - 然后**每执行一次宏任务队列的回调函数就再回到微任务队列看有没有新进来的微任务**