JavaScript是单线程的，只有当上一个任务完成之后才会继续完成下一个任务,如果前一个任务耗时很长，后一个任务就不得不一直等着。于是，所有任务可以分成两种，一种是同步任务（synchronous），另一种是异步任务（asynchronous）。

###### **同步任务**

在主线程上排队执行的任务，只有前一个任务执行完毕，才能执行后一个任务。

###### **异步任务**

不进入主线程、而进入"任务队列"（task queue）的任务，只有"任务队列"通知主线程，某个异步任务可以执行了，该任务才会进入主线程执行。

**Eventloop机制：**

1. 所有同步任务都在主线程上执行，形成一个执行栈。
2. 主线程之外，还存在一个"任务队列"。只要异步任务有了运行结果，就在"任务队列"之中放置一个事件。
3. 一旦"执行栈"中的所有同步任务执行完毕，系统就会读取"任务队列"，看看里面有哪些事件。那些对应的异步任务，于是结束等待状态，进入执行栈，开始执行。
4. 主线程不断重复上面的第三步。  
   主线程从任务队列中读取事件,这个过程是不断循环的,所以整个的运行机制称为event loop。

**Webapi：**将异步的任务放到api中，然后等待同步代码执行完就分配给宏任务或者微任务。

**宏任务和微任务**：

宏任务:每次执行栈执行的代码就是一个宏任务（包括每次从事件队列中获取一个事件回调并放到执行栈中执行,每一个宏任务会从头到尾将这个任务执行完毕，不会执行其它）包括整体代码script，setTimeout，setInterval  
微任务:是在当前任务队列执行结束后立即执行的任务包括Promise，process.nextTick，微任务执行完之后才会到宏任务中取出一个任务来执行。

