黑客派

说说 Event Loop

作者：zjhch123
原文链接：https://hacpai.com/article/1522152948738
来源网站：黑客派

说说Event Loop

最近浏览了很多前端的面试题，发现绝大多数题目里都有问到Event Loop。正好我收集整理了部分相的信息，于是也来说说我理解的Event Loop。

从单线程说起

为什么会有Event Loop？这就要从Javascript的特点“单线程”开始说起了。

单线程是什么？意思就是在一个时间内程序只能做一件事。很多人都用过Java或者C++之类的语言，定能体会到这些语言的多线程带来的很多便捷性。然而，Javascript在设计之初的定位是用来处理用交互以及操作DOM，如果Javascript也设计成多线程，势必会带来很复杂的同步问题。

既是单线程又是异步

单线程的JS中，所有的任务都要排队，只有等前一个任务执行完毕才会执行后一个任务。

那所谓的异步又是怎么回事呢？

在单线程的Javascript中，涉及到大IO操作的任务，我们都可以为其注册回调函数：当程序执行到IO作时，主线程将这个操作挂起，继续执行接下来的操作。等到IO操作完成之后，再将挂起的任务继续行下去。

这个注册了回调函数的任务，我们可以叫它“异步任务”。

任务队列(EventQueue)

任务队列是一个队列，具有先进先出的特点。也就是说，先被加入任务队列的任务，会优先被主线程取。

当一个异步任务完成之后，JS就会触发某一指定事件(比如说onload, onerror)，则其所指向的函数(调函数)就会被加入到任务队列中。

可以尝试理解以下代码。

```
const img = document.createElement('img')
img.src = 'https://image.hduzplus.xyz/image/1507523489652.jpg'
img.onload = function() {
  console.log(1)
}
const timeoutFunc = function() {
  console.log(3)
}
setTimeout(timeoutFunc, 0)
console.log(2)

// 执行结果：2 3 1
```

异步任务和同步任务

在JS中，任务分为两种，一种是上文提到的异步任务，另一种是同步任务。

同步任务意思就是在主线程上依此执行的任务。

异步任务就是进入任务队列的任务。只有等其执行完毕，任务队列通知主线程后，这个异步任务才会

入到主线程执行。

异步执行机制

1. 所有的同步任务在主线程上依此执行；
2. 异步任务执行完成后，相应事件的回调函数被加入到任务队列中；
3. 主线程上的同步任务执行完毕后，将任务队列内的任务依此移入主线程执行；
4. 重复以上3步。

```
// 伪代码
while(true) {
  // 执行主线程操作
  while(p = eventQueue.out()) {
    // process p
  }
}
```

Event Loop

主线程循环的从异步队列中读取事件，这个过程其实就是Event Loop。

接下来可以看之前的代码。

主线程顺序执行，遇到异步任务则将其挂起，等待异步任务执行完成后再将其回调加入主线程。

所以先输出主线程中的'2'

图片load事件比setTimeout事件执行的慢，所以setTimeout的回调被先加入任务队列。所以输出'3'

最后输出'1'

扩展：异步任务是如何在单线程中执行的？

事实上，JS的单线程指的是语言层面的单线程。在浏览器中我们执行某个ajax异步操作，这项异步操是由浏览器完成的。我们注册的回调函数，实际上是向浏览器提交了这个回调函数。当异步操作完成后，浏览器将该回调函数传入到执行上下文中的任务队列内。