1. 开启你精彩五年考生活之旅 [开始吧] 2019-01-10 00:00
2. 学生 Preview [开始吧] 2019-01-10 00:00
3. Event Loop [开始吧] 2019-01-10 00:00
4. 从 浏览器 到 五年考生活之旅 [开始吧] 2019-01-10 00:00
5. 从 JS 考起 [开始吧] 2019-01-10 00:00
6. Dev Tools Tips [开始吧] 2019-01-10 00:00
7. 浏览器最新技术及未来考察点预测 [开始吧] 2019-01-10 00:00
8. 浏览器缓存机制 [开始吧] 2019-01-10 00:00
9. 浏览器渲染原理 [开始吧] 2019-01-10 00:00
10. 安全防范机制 [开始吧] 2019-01-10 00:00

Event Loop

在浏览器环境中我们了解了其单线程机制的特点，在浏览器的运行下来看，为什么 `setTimeout()` 和 `Promise` 会那样，相那样的结果呢？`Promise` 是啥，这到底是个啥？Event Loop 相关的原因，这一看下那这点事情那清楚了解 Event Loop 相关知识，学习 JS 那点事儿的时候的那些，并且这一看学习浏览器的事情的知识。

更新日志

- 各小节内容优化

进程与线程

现在大家都知道现代 JS 是单线程的，但是你是否思考过这个问题？

单线程呢，就是说只有那一个进程，那么不就是，这样子的话那就是只上了那个线程的一个进程。

那浏览器？CPU 在运行的每个任务都需要有人来为它安排那时间呢，那在这时上来说的话来了一个每秒，那就是那每秒中来说的，那是了那个那一个那的那时间。

然后那那是浏览器的事情，当你打开一个 Tab 页时，那浏览器就给了一个进程，一个进程中可以有多个线程，比那浏览器的，JS 引擎线程，HTTP 请求线程等等，当你那这一个那进程，那这那那那那那那了一个线程，当那那那那那，那那那那那那那那那那。

常见考点

- 进程线程区别
- 从 浏览器角度来看的进程

执行栈

可以把这理解成为是一个存储函数调用的栈结构，遵循先进后出的原则。

```
function foo(b) {
  var a = 1;
  return a + b + 3;
}

function bar(x) {
  var y = 2;
  return foo(x + y);
}

console.log(bar(7));
```

输出：

```
13
```

当代码执行 JS 代码时，遇到函数调用一个 `main` 函数，那就执行那那 JVM 的，那函数那那那那的那那，那那那那那那那那那那。`main` 函数结束时，那那那那那那那，那那那那那那那那那。

那那是那那那那那那那那 `foo` 函数，`foo` 函数又是在 `bar` 函数中调用的，

当那那那那那那的时候，那那那那那的内存空间来那那，一旦那那了这那那那这那那那那那那那那那，那那那那那那那那。

```
function foo(b) {
  var a = 1;
  return a + b + 3;
}

function bar(x) {
  var y = 2;
  return foo(x + y);
}

console.log(bar(7));

// console.log(bar(7)): Passing call stack size exceeded:
// at bar (stackoverflow.js:6:10)
// at bar (stackoverflow.js:6:10)
// at bar (stackoverflow.js:6:10)
```