**setTimeout使用注意事项**

在 Chrome 中除了正常使用的消息队列之外，还有另外一个消息队列，这个队列中维护了需要延迟执行的任务列表，包括了定时器和 Chromium 内部一些需要延迟执行的任务。所以当通过 JavaScript 创建一个定时器时，渲染进程会将该定时器的回调任务添加到延迟队列中。

具体来说，是一个hashmap结构，等到执行这个结构的时候，会计算hashmap中的每个任务是否到期了，到期了就去执行，直到所有到期的任务都执行结束，才会进入下一轮循环。所以延迟队列和正常的消息队列是不冲突的。

**当前任务执行时间过久，会延迟定时任务的执行**

通过 setTimeout 设置的回调任务被放入了消息队列中并且等待下一次执行，这里并不是立即执行的；要执行消息队列中的下个任务，需要等待当前的任务执行完成，由于当前这段代码要执行 5000 次的 for 循环，所以当前这个任务的执行时间会比较久一点。这势必会影响到下个任务的执行时间。

function bar() {

/\* \*/

}

function foo() {

setTimeout(bar, 0)

for (let i = 0; i < 5000; i++) {

//do something

}

}

**setTimeout嵌套调用，系统会设置最短时间间隔为4毫秒**

在 Chrome 中，定时器被嵌套调用 **5** 次以上，系统会判断该函数方法被阻塞了，如果定时器的调用时间间隔小于 4 毫秒，那么浏览器会将每次调用的时间间隔设置为 4 毫秒。

下面是Chromium实现4毫秒延迟的代码：

static const int kMaxTimerNestingLevel = 5;

// Chromium uses a minimum timer interval of 4ms. We'd like to go

// lower; however, there are poorly coded websites out there which do

// create CPU-spinning loops. Using 4ms prevents the CPU from

// spinning too busily and provides a balance between CPU spinning and

// the smallest possible interval timer.

static constexpr base::TimeDelta kMinimumInterval = base::TimeDelta::FromMilliseconds(4);

base::TimeDelta interval\_milliseconds =

std::max(base::TimeDelta::FromMilliseconds(1), interval);

if (interval\_milliseconds < kMinimumInterval &&

nesting\_level\_ >= kMaxTimerNestingLevel)

interval\_milliseconds = kMinimumInterval;

if (single\_shot)

StartOneShot(interval\_milliseconds, FROM\_HERE);

else

StartRepeating(interval\_milliseconds, FROM\_HERE);

**未激活的页面，setTimeout执行最小间隔是1000毫秒**

未被激活的页面中定时器最小值大于1000毫秒，也就是说，如果当前标签页处于未激活的状态，那么定时器最小的时间间隔是1000毫秒，目的是为了**优化后台页面的加载损耗**以及**降低耗电量**。

**延迟执行时间有最大值**

Chrome、Safari、Firefox 都是以 32 个 bit 来存储延时值的，**32bit** 最大只能存放的数字是 2147483647 毫秒，这就意味着，如果 setTimeout 设置的延迟值大于 2147483647 毫秒（大约 24.8 天）时就会溢出，这导致定时器会被立即执行。

function showTime() {

console.log(new Date())

}

let timer = setTimeout(showTime, 2147483648); // 立即执行

**setTimeout回调函数中的this指向**

如果被 setTimeout 推迟执行的回调函数是某个对象的方法，那么该方法中的 this 关键字将指向全局环境，而不是定义时所在的那个对象。

**总结**

* setTimeout嵌套调用5次以上，时间间隔最小为4毫秒；
* 在未激活页面运行，最小时间间隔为1000毫秒；
* 最大时间间隔为32个bit可存储的值，也就是2147483647 毫秒（大约 24.8 天）。