Node.js v6.10.0 文档

返回文档首页

目录

定时器

查看英文版 / 参与翻译

稳定性: 3 - 锁定的

`timer` 模块暴露了一个全局的 API，用于在某个未来时间段调用调度函数。 因为定时器函数是全局的，所以使用该 API 无需调用 `require('timers')`。

Node.js 中的计时器函数实现了与 Web 浏览器提供的定时器类似的 API，除了它使用了一个不同的内部实现，它是基于 Node.js 事件循环构建的。

Immediate 类

查看英文版 / 参与翻译

该对象是内部创建的，并从 `setImmediate()` 返回。 它可以传给 `clearImmediate()` 以便取消预定的动作。

Timeout 类

查看英文版 / 参与翻译

该对象是内部创建的，并从 `setTimeout()` 和 `setInterval()` 返回。 它可以传给 `clearTimeout()` 或 `clearInterval()` 以便取消预定的动作。

默认情况下，当使用 `setTimeout()` 或 `setInterval()` 预定一个定时器时，只要定时器处于活动状态，Node.js 事件循环就会继续运行。 每个由这些函数返回的 `Timeout` 对象都导出了可用于控制这个默认行为的 `timeout.ref()` 和 `timeout.unref()` 函数。

timeout.ref()

查看英文版 / 参与翻译

新增于: v0.9.1

调用时，只要 `Timeout` 处于活动状态就要求 Node.js 事件循环不要退出。 多次调用 `timeout.ref()` 没有效果。

注意：默认情况下，所有 `Timeout` 对象都是 "ref'd" 的，通常不需要调用 `timeout.ref()`，除非之前调用了 `timeout.unref()`。

返回 `Timeout` 的一个引用。

timeout.unref()

查看英文版 / 参与翻译

新增于: v0.9.1

当调用时，活动的 `Timeout` 对象不要求 Node.js 事件循环保持活动。 如果没有其他活动保持事件循环运行，则进程可能在 `Timeout` 对象的回调被调用之前退出。 多次调用 `timeout.unref()` 没有效果。

注意：调用 `timeout.unref()` 会创建一个内部定时器，它会唤醒 Node.js 的事件循环。 创建太多这类定时器可能会对 Node.js 应用程序的性能产生负面影响。

返回对 `Timeout` 的一个引用。

预定定时器

查看英文版 / 参与翻译

Node.js 中的计时器是一种会在一段时间后调用给定的函数的内部构造。 定时器函数何时被调用取决于用来创建定时器的方法以及 Node.js 事件循环正在做的其他工作。

setImmediate(callback[, ...args])

查看英文版 / 参与翻译

新增于: v0.9.1

- `callback` <Function> 在 Node.js 事件循环的当前回合结束时要调用的函数。
- `...args` <any> 当调用 `callback` 时要传入的可选参数。

预定立即执行的 `callback`，它是在 I/O 事件的回调之后、在使用 `setTimeout()` 和 `setInterval()` 创建的计时器之前被触发。 返回一个用于 `clearImmediate()` 的 `Immediate`。

当多次调用 `setImmediate()` 时，`callback` 函数会按照它们被创建的顺序依次执行。 每次事件循环迭代都会处理整个回调队列。 如果一个立即定时器是被一个正在执行的回调排入队列的，则该定时器直到下一次事件循环迭代才会被触发。

如果 `callback` 不是一个函数，则抛出 `TypeError`。

setInterval(callback, delay[, ...args])

查看英文版 / 参与翻译

新增于: v0.0.1

- `callback` <Function> 当定时器到点时要调用的函数。
- `delay` <number> 调用 `callback` 之前要等待的毫秒数。
- `...args` <any> 当调用 `callback` 时要传入的可选参数。

预定每隔 `delay` 毫秒重复执行的 `callback`。 返回一个用于 `clearInterval()` 的 `Timeout`。

当 `delay` 大于 `2147483647` 或小于 `1` 时，`delay` 会被设为 `1`。

如果 `callback` 不是一个函数，则抛出 `TypeError`。

setTimeout(callback, delay[, ...args])

查看英文版 / 参与翻译

新增于: v0.0.1

- `callback` <Function> 当定时器到点时要调用的函数。
- `delay` <number> 调用 `callback` 之前要等待的毫秒数。
- `...args` <any> 当调用 `callback` 时要传入的可选参数。

预定在 `delay` 毫秒之后执行的单次 `callback`。 返回一个用于 `clearTimeout()` 的 `Timeout`。

`callback` 可能不会精确地在 `delay` 毫秒被调用。 Node.js 不能保证回调被触发的确切时间，也不能保证它们的顺序。 回调会在尽可能接近所指定的时间上调用。

注意：当 `delay` 大于 `2147483647` 或小于 `1` 时，`delay` 会被设为 `1`。

如果 `callback` 不是一个函数，则抛出 `TypeError`。

取消定时器

查看英文版 / 参与翻译

`setImmediate()`、`setInterval()` 和 `setTimeout()` 方法每次都会返回表示预定的计时器的对象。 它们可用于取消定时器并防止触发。

clearImmediate(immediate)

查看英文版 / 参与翻译

新增于: v0.9.1

- `immediate` <Immediate> 一个 `setImmediate()` 返回的 `Immediate` 对象。

取消一个由 `setImmediate()` 创建的 `Immediate` 对象。

clearInterval(timeout)

查看英文版 / 参与翻译

新增于: v0.0.1

- `timeout` <Timeout> 一个 `setInterval()` 返回的 `Timeout` 对象。

取消一个由 `setInterval()` 创建的 `Timeout` 对象。

clearTimeout(timeout)

查看英文版 / 参与翻译

新增于: v0.0.1

- `timeout` <Timeout> 一个 `setTimeout()` 返回的 `Timeout` 对象。

取消一个由 `setTimeout()` 创建的 `Timeout` 对象。