单线程：主要用途是与用户互动

同步任务：主线程上排队执行的任务

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

任务队列：先进先出的数据结构、事件的队列

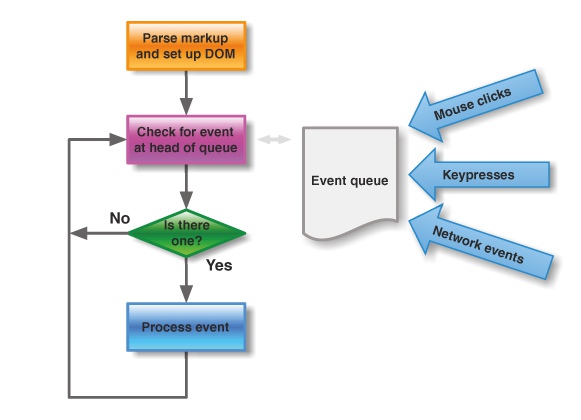
event loop：主线程从"任务队列"中读取事件，这个过程是循环不断的。

（1）所有同步任务都在主线程上执行，形成一个[执行栈](http://www.ruanyifeng.com/blog/2013/11/stack.html)（execution context stack）。

（2）主线程之外，还存在一个"任务队列"（task queue）。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。



Stack：

**含义一：数据结构**

第一种含义是一组数据的存放方式，特点为LIFO，即后进先出

**含义二：代码运行方式**

stack的第二种含义是["调用栈"](http://en.wikipedia.org/wiki/Call_stack)（call stack），表示函数或子例程像堆积木一样存放，以实现层层调用。

**含义三：内存区域**

stack的第三种含义是[存放数据的一种内存区域](http://en.wikipedia.org/wiki/Stack-based_memory_allocation)，stack是有结构的，每个区块按照一定次序存放，可以明确知道每个区块的大小；heap是没有结构的，数据可以任意存放。因此，stack的寻址速度要快于heap。