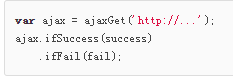
# Promise

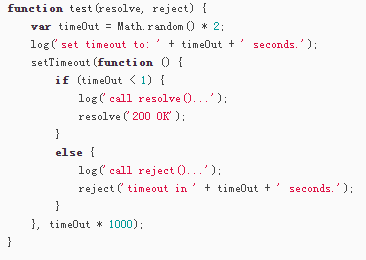
在JavaScript的世界中，所有代码都是单线程执行的。

异步执行可以用回调函数实现。



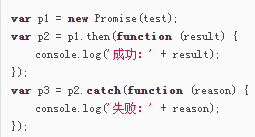
链式写法，先统一执行AJAX逻辑，不关心如何处理结果，然后，根据结果是成功还是失败，在将来的某个时候调用success或者fail函数。

Promise实例：生成一个0-2之间的随机数，如果小于1，则等待一段时间后返回成功，否则返回失败。



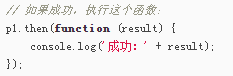
Test的两个参数都是函数，test函数值关心自身的逻辑，并不关心具体的resolve和reject将如何处理结果。

我们可以使用promise对象来执行，并在将来某个时刻获得成功或失败的结果。

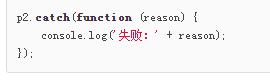


变量p1是一个promise对象，他负责执行test函数。由于test函数是异步执行的，当test函数执行成功时，我们告诉promise对象：

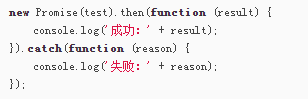
//由于成功，执行这个函数



当test函数执行失败时，我们告诉promise对象：

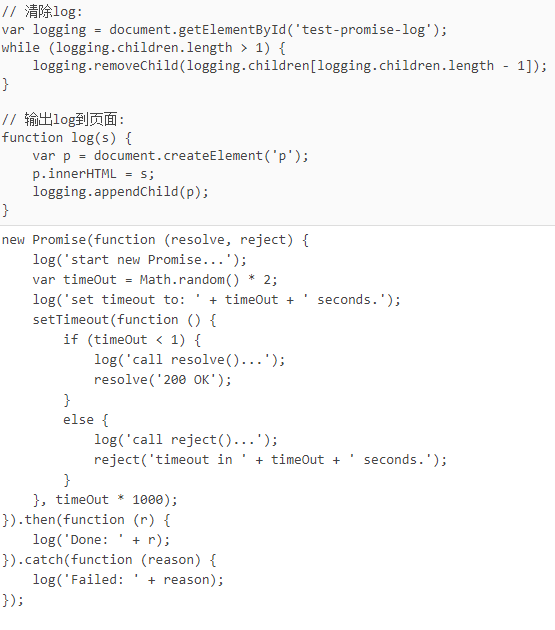


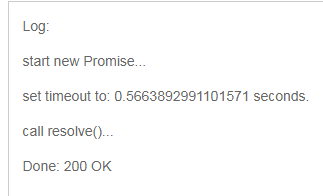
串联起来



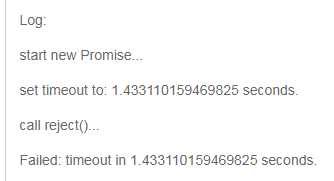
## 测试promise是如何异步执行的

Promise最大的好处是在异步执行的流程中，把执行代码和处理结果的代码清晰地分离了。

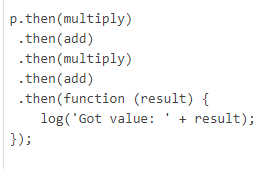




或者



## Promise串行执行



## Promise并行执行

