#### ES2019提案预览

##### 之异步交互

摘要

Angular2特性——异步交互

@link <https://github.com/tc39/proposal-promise-any>

@link <https://github.com/tc39/proposal-promise-allSettled>

@link <https://github.com/tc39/proposal-top-level-await>

正文

Promise.any()

Promise.any是一个新的异步控制方法。如果一组中有一个Promise成功执行，那么返回该Promise执行结果。如果所有Promise都执行失败，那么返回一组错误原因。

Promise.allSettled()

There are [four main combinators in the Promise landscape](https://v8.dev/features/promise-combinators).

| name | description |  |
| --- | --- | --- |
| Promise.allSettled | does not short-circuit | this proposal |
| Promise.all | short-circuits when an input value is rejected | added in ES2015 ✅ |
| Promise.race | short-circuits when an input value is settled | added in ES2015 ✅ |
| Promise.any | short-circuits when an input value is fulfilled | [separate proposal](https://github.com/tc39/proposal-promise-any) |

These are all commonly available in userland promise libraries, and they’re all independently useful, each one serving different use cases.

A common use case for this combinator is wanting to take an action after multiple requests have completed, regardless of their success or failure. Other Promise combinators can short-circuit, discarding the results of input values that lose the race to reach a certain state. Promise.allSettled is unique in always waiting for all of its input values.

Promise.allSettled returns a promise that is fulfilled with an array of promise state snapshots, but only after all the original promises have settled, i.e. become either fulfilled or rejected.

Promise.allSettled()相当于Observer.forkJoin().

Top Level await

顶级等待使模块可以充当大型异步功能：使用顶级等待，ECMAScript模块（ESM）可以等待资源，从而导致其他导入模块的模块在开始执行其主体之前等待。

Java中可用并发工具类CountDownLatch（java.util.concurrent）实现类似的功能。

□