Promise

目录

- UnhandledPromiseRejectionWarning

UnhandledPromiseRejectionWarning

如果 Promise 运行过程中抛出错误，或者状态变为 `rejected`，但是没有相应的处理代码，那么 Node 会抛出一个警告 `UnhandledPromiseRejectionWarning`。

```
new Promise(function (resolve, reject) {
  reject('message');
  console.log('hello');
})
// hello
// UnhandledPromiseRejectionWarning: message
```

上面代码中，Promise 变为 `rejected` 状态，但是没有处理的代码，导致抛出警告 `UnhandledPromiseRejectionWarning`。由于抛出的是警告，而不是错误，所以不影响 `hello` 的输出。请跟下面的代码比较一下。

```
new Promise(function (resolve, reject) {
  throw new Error('message');
  console.log('hello');
})
// UnhandledPromiseRejectionWarning: Error: message
```

上面代码中，Promise 内部抛出错误，没有得到处理，所以会有警告 `UnhandledPromiseRejectionWarning`。Promise 内部的错误阻止 Promise 内部的代码继续向下运行，所以不会输出 `hello`。

消除这个警告的方法很简单，就是为 Promise 加上错误处理代码，即 `catch` 代码块。

```
new Promise(function (resolve, reject) {
  reject('message');
  console.log('hello');
}).catch(function (err) { console.log (err) })
// hello
// message
```

上面代码中，由于 `catch` 代码块的存在，所以不会抛出警告。

async 函数里面使用 Promise，也要放在 `try/catch` 代码块之中。

```
const p = new Promise(function (resolve, reject) {
  reject('message');
});

(async function () {
  try {
    await p;
  } catch (err) {
    console.log(err);
  }
})();
// message
```

上面代码中，如果 Promise 没有放在 `try/catch` 代码块，就会抛出警告 `UnhandledPromiseRejectionWarning`。

Node 还提供 `process` 对象的 `unhandledRejection` 事件，在抛出警告 `UnhandledPromiseRejectionWarning` 之前，就会触发这个事件。如果设置了这个事件的处理代码，就不会抛出警告 `UnhandledPromiseRejectionWarning`。

```
new Promise(function (resolve, reject) {
  reject('message');
});

process.on('unhandledRejection', function (err) {
  console.log(err);
});
// message
```

Node 开发团队打算废除 `UnhandledPromiseRejectionWarning`，以后如果没有相应的处理代码，Promise 内部抛错或者变为 `rejected` 状态，会导致 Node 进程终止执行，并且有一个非零的返回码。所以应该养成习惯，只要有 Promise ，就要部署失败情况下的代码。

□□□ #1

□ □□□ □□□ 2019-03-05 11:24:38 UTC

□ □□□ □□□ 2019-03-05 11:24:38 UTC