【ES6】非同步 與 promise

非同步

```
function getData() {
  setTimeout(() => {
    console.log('... 已取得遠端資料');
  }, 0);
}
// 請問取得資料的順序為何
function init() {
  console.log(1);
  getData();
  console.log(2);
}
init();
////////////// result
1
2
... 已取得遠端資料
```

premise 使程式流程同步

```
const promiseSetTimeout = (status) => {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      if (status) {
        resolve('promiseSetTimeout 成功')
      } else {
        reject('promiseSetTimeout 失敗')
      }
    }, 0);
  })
}

// #2-1 基礎運用 (成功用 then 捕捉, 失敗用 catch捕捉)
// 呼叫成功
console.log(1)
promiseSetTimeout(1)
.then((res)=>{
    console.log(res);
})
.catch((err)=>{
    console.log(err);
})
console.log(2)
/** result
1
promiseSetTimeout 成功
2
*/

// 呼叫失敗
console.log(1)
promiseSetTimeout(2)
.then((res)=>{
    console.log(res);
})
.catch((err)=>{
    console.log(err);
})
console.log(2)
/** result
1
promiseSetTimeout 失敗
2
```

```
*/
```

promise 串接（不使用巢狀寫法)

```
promiseSetTimeout(1)
.then((res)=>{
    console.log(res);

    // 使用return 發起第二次非同步
    return promiseSetTimeout(1);
})
// 第二次非同步成功
.then((res) => {
    console.log("第二次非同步成功");
})
```

修訂版本 #2
★由 treeman 建立於 27 ﷽@﷽ 2023 02:38:12
✎由 treeman 更新於 5 ﷽﷽ 2023 10:14:59