

追求优雅的代码


让工作更具挑战，让编码充满乐趣

<https://github.com/zongzi531>

「优雅」

简洁 · 自然 · 易读

用代码来阐述



```
1 interface Item {  
2     value: boolean;  
3 }  
4  
5 const list: Array<Item> = []  
6  
7 list.map(item => {  
8     item.value = true  
9 })  
10  
11 list.forEach(item => {  
12     item.value = true  
13 })  
14  
15 list.map(item => ({ ...item, value: true })))
```

好的注释



```
1 /**
2  * Item 开关按钮
3  * @prototype {Boolean} value 是否开启
4  */
5 interface Item {
6     value: boolean;
7 }
```

巧用命名空间



```
1 interface Switch {  
2     enabled: boolean;  
3 }  
4  
5 const switches: Array<Switch> = []  
6  
7 switches.forEach(switch => {  
8     switch.enabled = true  
9 })
```

专一 · 纯粹



```
1 interface Switch {  
2     enabled: boolean;  
3 }  
4  
5 const switches: Array<Switch> = []  
6  
7 const enabledSwitchs = () => {  
8     switches.forEach(switch => {  
9         switch.enabled = true  
10    })  
11 }  
12  
13 enabledSwitchs()
```



「质量」

高性能

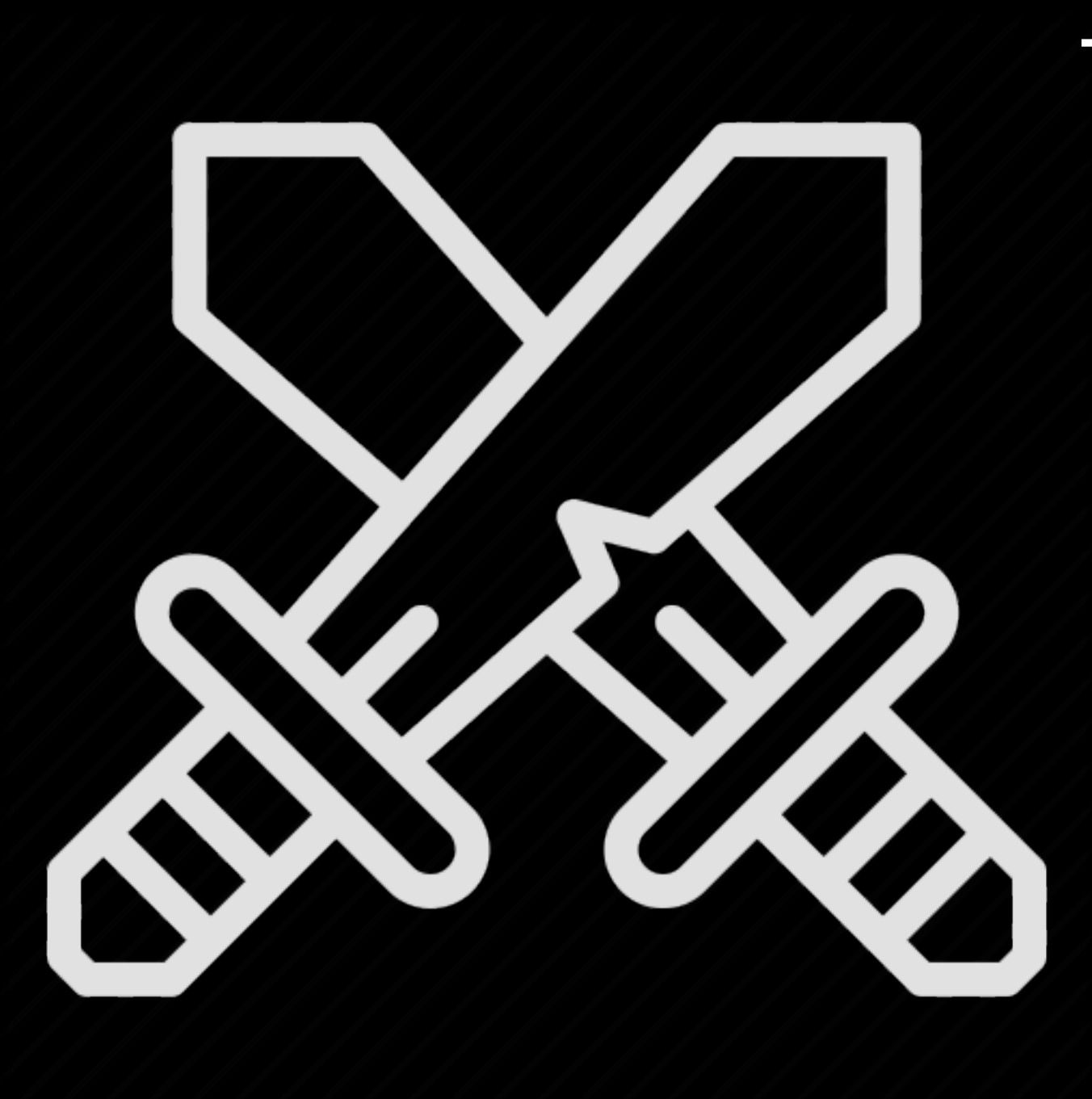
高性能



有名的较量

for in

for of




forEach

map

while

避免重复执行



```
1 export const doSoming = () => {
2     if (process.env.NODE_ENV === 'production') {
3         return proDoSoming()
4     }
5     return devDoSoming()
6 }
7
8 /* 另一种 */
9 export let doSoming: () => void
10
11 const initDoSoming = () => {
12     if (process.env.NODE_ENV === 'production') {
13         doSoming = proDoSoming
14     }
15     doSoming = devDoSoming
16 }
17
18 initDoSoming()
```

空间换时间



```
1 const cache: Record<string, number> = {}  
2  
3 export const memoizeAdd = (a: number, b: number) => {  
4     const key = [a, b].join()  
5     if (cache[key]) {  
6         return cache[key]  
7     }  
8     cache[key] = a + b  
9     return cache[key]  
10 }
```

空间换时间



```
1 const cache: Record<string, number> = {}  
2  
3 export const memoizeAdd = (a: number, b: number) => {  
4     const key = [a, b].join()  
5     if (Object.hasOwnProperty.call(cache, key)) {  
6         return cache[key]  
7     }  
8     cache[key] = a + b  
9     return cache[key]  
10 }
```



易维护

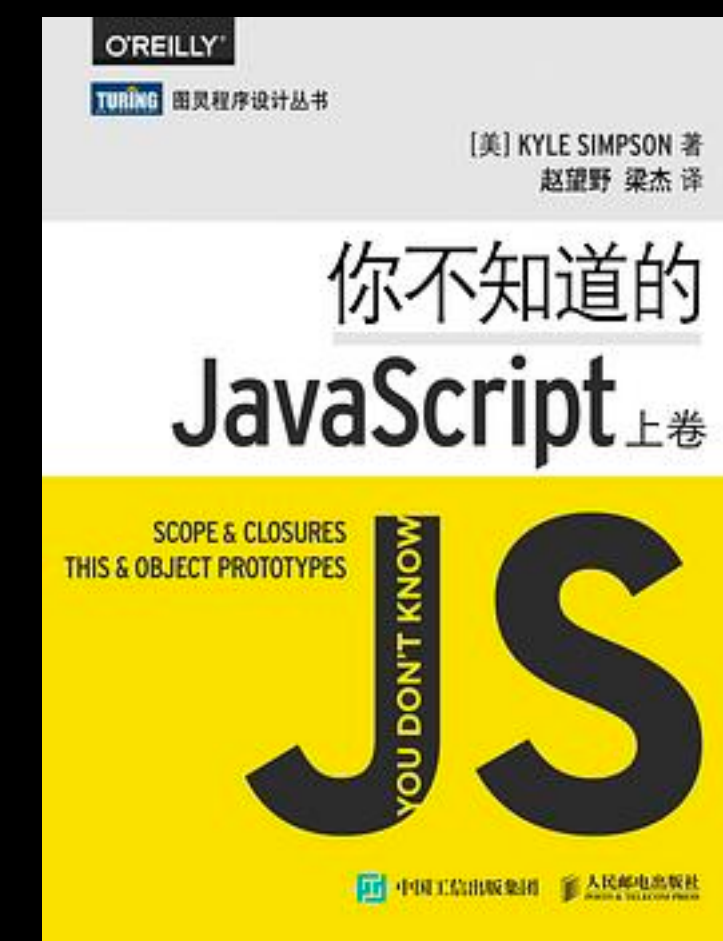
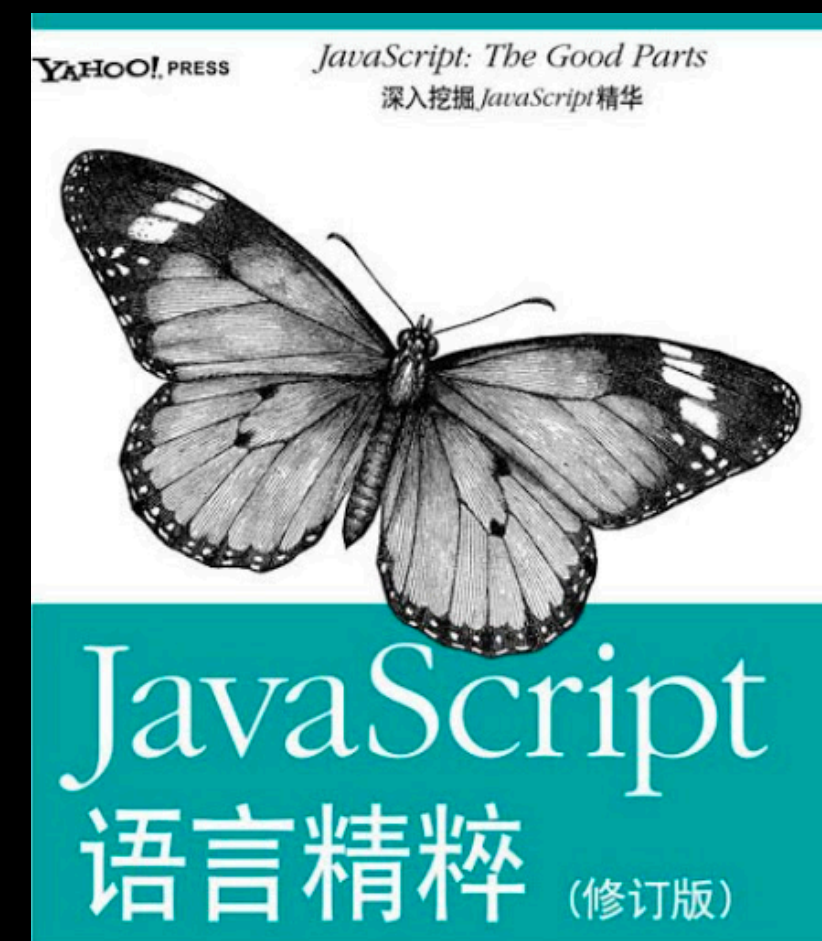
设计模式 · 架构设计

「追求」

多看，多尝试

多看，多尝试

GitHub



ESLint

ESLint



```
1 👎 /*eslint semi: ["error", "always"]*/  
2  
3 var name = "ESLint"  
4  
5 object.method = function() {  
6     // ...  
7 }
```



```
1 👍 /*eslint semi: "error"*/  
2  
3 var name = "ESLint";  
4  
5 object.method = function() {  
6     // ...  
7 };
```

ESLint



```
1 🙄/*eslint no-prototype-builtins: "error"*/
2
3 var hasBarProperty = foo.hasOwnProperty("bar");
4
5 var isPrototypeOfBar = foo.isPrototypeOf(bar);
6
7 var barIsEnumerable = foo.propertyIsEnumerable("bar");
```



```
1 👍/*eslint no-prototype-builtins: "error"*/
2
3 var hasBarProperty = Object.prototype.hasOwnProperty.call(foo, "bar");
4
5 var isPrototypeOfBar = Object.prototype.isPrototypeOf.call(foo, bar);
6
7 var barIsEnumerable = {}.propertyIsEnumerable.call(foo, "bar");
```


ESLint



```
1 🙅 /*eslint use-isnan: "error"*/  
2  
3 if (foo == NaN) {  
4     // ...  
5 }  
6  
7 if (foo != NaN) {  
8     // ...  
9 }
```



```
1 👍 /*eslint use-isnan: "error"*/  
2  
3 if (isNaN(foo)) {  
4     // ...  
5 }  
6  
7 if (!isNaN(foo)) {  
8     // ...  
9 }
```

保持一颗优雅的心

保持一颗优雅的心

- 知识运用，持续优化
- 活跃关注，不断提升

「目的」

更强

总之，
坚持你的坚持。

提问环节

谢谢！