# 编写可读代码的艺术

1. 容易理解的代码

紧凑和直白之间的选择

可读性，必须容易理解

代码要少但不是越少越好

好名字当注释 专业词 避免宽泛词，表现力 近义词典 tmp i j a b item

具体代替抽象，具体名词，具体动词

名字信息量 带单位 类型 限制 危险

长度和下划线 长度和使用状况有关，如果局部调用可以短 全局使用最好写明白

单词补全 少用缩写 关于特殊格式名字 如私有变量\_\_ 常量大写NAME jq返回值用$做前缀

避免名字歧义 二义性词语 max min begin end first last 布尔命名 has is can should disable able

不要违背常用规范，函数和通常理解不同

不要被误解

**审美**

相似的代码应该对齐，换行

列对齐，对齐每个单词头部和等号，中间的用空格填充 如果是多个逗号分隔，可以对齐逗号和每个单词的头部

可以调整代码顺序实现对齐 顺序可以是表单，表数据顺序 也可以是字母顺序等

代码分组 按组注释对齐 代码分段几行一分再打上注释

注释

不用给一眼就能看出的功能注释 比如标准库函数的调用 自定义函数不要写无用注释

函数和变量名看不懂不要注释，而应该写好名字

记录思想，见解 为瑕疵注释

常量注释 可能的陷阱

全局注释，表示数据交互，类之间的关系，函数之间的调用

注释紧凑，不要太多行 精确描述函数功能 输入/输出示例类似ACM 参数注释

简化循环和逻辑

控制流 if (i<10) 左边是变量，右边是常量

避免出错把 false==a反写 已经过时 尤达表示法

If else 顺序 正逻辑 负逻辑！ 先正后负，先简单后复杂 先有趣或者危险

三目少用

避免do while循环

提前return 多出口

减少多层嵌套 if else if 提前返回 return

减少循环嵌套 continue

拆分大段的表达式

子表达式赋值给变量 比如函数调用 getElementById 方便复用

关于布尔 交并补 使用反义等价写法 把一行条件拆成多行

拆分复杂逻辑

减少不必要的变量

无用的临时变量，取代的表达式不够长，而且变量只用了一次

中间结果

控制流变量，用外部变量来决定是否循环以及循环内容

缩小作用域 全局改局部， 成员变量改方法变量

变量可以只写一次，函数式编程 const final 常量

抽取子问题，大问题拆成小问题，抽取到单独的函数 主问题调用子问题函数

工具代码 如交换字符串应该被抽取 封装

创建通用函数，形成独立库

包装已有函数，减少参数等，使使用更方便 重塑

代码、函数 一次只做一件事

拆分任务，分步

自然语言描述代码，使代码变得清晰，不要违背人的直觉，写代码就像文章一样

少写代码

削减需求，解决更简单的问题

缩小代码库，使用工具函数复用，拆分子项目

舍得删除无用代码

熟悉库，尽量用标准库解决问题