??

??????

1. 好的设计从命名开始
2. 研究表明，最优秀的设计师会设计出更快、更小、更简单的结构，而且设计过程也更轻松。伟大的结构和一般的结构之间差了一个数量级 ——Fred Brooks,Jr.,1986

??

1. 过度点缀
   1. 明确所在的前提，背景，范围
      1. 在 test 文件夹里面的 xxx_test.cpp 里面定义一个 class xxx_test，这里的文件名和类的test是多余的
      2. 在 cmake里面定义一个测试可以命名成 xxx_test ，这里的 test不多余，因为平行的有xxx
2. 过度设计
   1. 定义太多不必要的抽象、名词
      1. 会给使用者 /阅读者造成难度和压力
      2. 命名空间、类（面向对象）的组织要有一定的层级和明确的结构关系
   2. 统一前后的名词
      1. 同一种事物 /抽象 /表示，在所有的地方使用一个名字，大小写和命名规则也一致
         1. cpu_state cpu_states cpu_s c_state cs
3. 不良设计
   1. 歧义
   2. 片面
      1. class KimiSoul: # The soul of Kimi Code CLI. 应该采用 Core， Base更能表示其作用，而不是采用 比喻性概念

??

1. 通过命名引导读者理解设计意图
   1. auto cpu53 = A53(); auto cpu45 = A45(); cpu53和 cpu45明确引导读者，这都是 cpu只不过对应不同的类型。
2. 多单词命名的合理排序
   1. BreakpointInsert BreakpointRemove BreakpointRemoveAll
   2. InsertBreakpoint RemoveBreakpoint RemoveAllBreakpoint
3. 代码 /函数 的位置选择
   1. #define Trace xxx 是 trace功能的一部分，虽然可以放到任意其他的地方，但是最好是所有的trace相关的代码放一起

???????????

1. 制定规则和培训

Revision #19
Created 2025-01-10 03:52:11 UTC by Colin
Updated 2026-04-29 07:33:54 UTC by Colin