Python 技术管理规范

版本：1.0

更新日期：2025-06-12

规范级别说明

* a（强制）：必须遵守，代码审查会拦截违反项。
* b（推荐）：强烈建议遵守，需团队审批豁免。
* c（允许）：根据场景灵活使用。

一、代码风格与格式

1. a 使用 4 个空格缩进（禁止 Tab 与空格混用）。

2. a 每行不超过 80 字符（URL、长导入除外）。

3. a 文件末尾保留一个空行。

4. b 运算符两侧、逗号后保留 1 个空格（`x = y + z`）。

5. a 使用双引号 `""` 定义字符串（与 JSON 保持一致）。

6. b 函数/类之间用 2 个空行分隔，类内方法用 1 个空行分隔。

7. a `import` 分组顺序：标准库 → 第三方库 → 本地模块（每组空一行）。

8. a 禁止通配符导入（`from module import \*`）。

二、命名规范

9. a 变量/函数名：小写 + 下划线（`calculate\_total`）。

10. a 类名：大驼峰（`ClassName`）。

11. a 常量：全大写 + 下划线（`MAX\_LENGTH`）。

12. a 保护成员：单下划线前缀（`\_internal\_var`）。

13. a 私有成员：双下划线前缀（`\_\_private\_var`）。

14. b 避免单字符变量名（循环索引 `i` 除外）。

15. a 模块名：简短小写（避免特殊字符）。

三、注释与文档

16. a 公共模块/类/函数必须包含 Docstring（遵循 Google 风格）。

17. b 复杂逻辑需添加行内注释（说明 为什么、 而非 是什么）。

18. a 修改代码时同步更新注释。

19. c 类型注解推荐用于公共接口（`def func(arg: int) -> str:`）。

20. b 废弃代码用 `# TODO(作者): 说明` 标记。

四、函数与类设计

21. a 函数长度 ≤ 50 行（超出需拆分）。

22. b 函数参数 ≤ 4 个（过多时改用 `dataclass` 或 `\*\*kwargs`）。

23. a 类长度 ≤ 300 行（超出需重构）。

24. b 优先组合而非继承。

25. a 禁止在 `\_\_init\_\_` 外初始化 动态属性（避免 `self.foo = None`）。

26. b 使用 `@property` 封装属性访问逻辑。

五、异常处理

27. a 捕获具体异常（禁止 `except:` 或 `except Exception`）。

28. a 异常消息需明确错误原因（`raise ValueError("Invalid ID format")`）。

29. b 使用上下文管理器管理资源（`with open() as f:`）。

30. a 自定义异常以 `Error` 结尾（`class ValidationError(Exception)`）。

六、性能与安全

31. b 字符串拼接用 `join` 替代 `+`（尤其循环内）。

32. a 禁止 `eval()` 或 `exec()`（动态代码执行需安全审查）。

33. b 查询数据库使用参数化（避免 SQL 注入）。

34. a 敏感信息禁止硬编码（使用环境变量）。

七、测试与维护

35. a 核心逻辑必须覆盖单元测试（覆盖率 ≥ 80%）。

36. b 测试用例名需描述行为（`test\_get\_user\_with\_invalid\_id\_fails`）。

37. a 使用 `pylint` + `flake8` 静态检查（CI 流程集成）。

38. b 使用 `black` 自动格式化代码。

八、工具与流程

39. a 依赖管理使用 `requirements.txt` + `pipenv`。

40. a 提交前运行本地测试（`pytest`）。

41. b 提交消息遵循语义化（`feat: Add user login API`）。

42. c 允许使用 Type Hint 提升可读性（非强制）。

九、其他

43. a 禁止可变对象作为默认参数（`def foo(a=[])` → `def foo(a=None)`）。

44. b 优先 `isinstance()` 检查类型（而非 `type()`）。

45. a 多进程共享资源用 `multiprocessing.Manager`。

46. c 允许使用 Walrus 运算符（`if (n := len(a)) > 10`）。

47. b 优先选择列表推导式（非 `map`/`filter`）。

48. a 跨文件共享常量定义在 `constants.py`。

附：参考来源

> - Google Python Style Guide

> - PEP 8 (Python 官方风格指南)

> -《Clean Code in Python》