Python代码规范管理文档

作者：罗皓天 2021141450210

文档概述

本文档基于PEP 8、Google Python Style Guide以及业界最佳实践，为Python项目制定统一的代码规范。规范分为三个级别：强制、推荐、允许，旨在提高代码质量、可读性和团队协作效率。

规范级别说明

- **强制（MUST）**：必须严格遵守，违反将导致代码审查不通过
- **推荐（SHOULD）**：建议遵守，有助于代码质量提升
- **允许（MAY）**：可选择性遵守，根据项目需要灵活应用

A. 强制规范（MUST）

A.1 代码风格

1. **缩进使用4个空格**，禁止使用Tab字符
2. **每行代码长度不得超过120字符**
3. **文件编码必须使用UTF-8**，文件头部添加编码声明
4. **导入语句必须独占一行**，禁止在同一行导入多个模块
5. **类名使用PascalCase命名**（如：`UserManager`）
6. **函数和变量名使用snake_case命名**（如：`get_user_info`）
7. **常量使用全大写字母加下划线**（如：`MAX_RETRY_COUNT`）
8. **私有属性和方法必须以单下划线开头**（如：`_private_method`）

A.2 代码结构

1. **所有公共函数和类必须包含docstring**
2. **导入顺序必须遵循：标准库 → 第三方库 → 本地模块**
3. **每个模块必须包含模块级docstring**
4. **函数参数个数不得超过6个**，超过时使用字典或数据类
5. **禁止使用**`from module import *`
6. **异常处理必须指定具体异常类型**，禁止使用裸露的except

A.3 安全规范

1. **禁止在代码中硬编码密码、密钥等敏感信息**
2. **必须验证所有外部输入**
3. **SQL查询必须使用参数化查询**，禁止字符串拼接
4. **文件操作必须使用上下文管理器**（with语句）

B. 推荐规范（SHOULD）

B.1 代码质量

1. **函数长度建议控制在50行以内**
2. **类的方法数量建议不超过20个**
3. **建议使用类型注解**（Type Hints）提高代码可读性
4. **建议使用列表推导式替代简单的for循环**
5. **建议使用enumerate()而非range(len())进行索引遍历**
6. **建议使用f-string进行字符串格式化**
7. **建议使用pathlib处理文件路径**而非os.path

B.2 设计模式

1. **建议遵循单一职责原则**，每个函数只做一件事
2. **建议使用工厂模式创建复杂对象**
3. **建议使用装饰器实现横切关注点**（如日志、性能监控）
4. **建议使用上下文管理器管理资源**
5. **建议使用生成器处理大数据集**

B.3 测试规范

1. **建议测试覆盖率达到80%以上**
2. **建议使用pytest作为测试框架**
3. **建议测试函数命名以test_开头**
4. **建议每个测试函数只测试一个功能点**

C. 允许规范（MAY）

C.1 高级特性

1. **可以使用lambda函数处理简单逻辑**
2. **可以使用魔法方法实现特殊功能**
3. **可以使用元类进行高级抽象**（需充分注释）
4. **可以使用异步编程处理I/O密集型任务**

C.2 工具使用

1. **可以使用black进行代码自动格式化**
2. **可以使用flake8进行代码风格检查**
3. **可以使用mypy进行静态类型检查**
4. **可以使用pre-commit进行提交前检查**

C.3 性能优化

1. **可以使用slots优化内存使用**
2. **可以使用functools.lru_cache进行结果缓存**
3. **可以使用multiprocessing处理CPU密集型任务**

规范执行

1. **代码审查**：所有代码提交前必须通过同行评审
2. **自动化检查**：CI/CD流程中集成代码风格检查
3. **定期培训**：每季度组织Python最佳实践培训
4. **文档更新**：规范文档每半年更新一次

参考资料

- PEP 8 — Style Guide for Python Code
- Google Python Style Guide
- The Zen of Python

作者：罗皓天 2021141450210

时间：2025年6月