版本控制规范

# 一、仓库结构与分支管理策略

- 主分支（main/master）：用于发布的稳定版本代码。  
- 开发分支（develop）：集成开发分支，所有功能合并到这里。  
- 功能分支（feature）：每个新功能一个分支，从 develop 分出，完成后合并回 develop。

# 二、提交规范

- 每次提交应保持原子性，完成一个功能点或修复一个问题。  
- 提交信息格式：  
 <类型>(<模块>): <简短描述>  
 类型示例：feat（新功能）、fix（修复）、docs（文档）、style（格式）、refactor（重构）等。  
- 提交信息应简洁明了，必要时在正文详细描述更改内容。

# 三、代码合并流程

- 通过 Pull Request (PR) 或 Merge Request (MR) 提交合并请求。  
- 合并时需保证本地代码已通过编译和测试。  
- 避免在主分支直接提交代码，所有变更通过分支合并完成。

# 四、冲突处理

- 遇到冲突时，优先与相关开发者沟通，确认合并方案。  
- 本地解决冲突后，重新运行测试，确保代码功能正常。  
- 合并前拉取最新代码，减少冲突概率。

# 五、版本标签

- 使用语义化版本号（Semantic Versioning）：主版本.次版本.修订号（如 v1.2.3）。  
- 每次发布版本，打标签并推送到远程仓库。