**SAM 项目团队本地开发环境统一规划**

**1. 操作系统**

* **统一版本：** 所有开发成员统一使用 Windows 10 操作系统，以减少因操作系统差异带来的兼容性问题。

**2. 版本控制工具**

* **统一版本：** 所有开发成员统一安装 Git 2.47 版本。
* **安装方式：** 建议从 Git 官网下载安装包进行安装，并按照默认设置进行安装。
* **配置：**
  + **全局用户名和邮箱：**

Bash

git config --global user.name "Your Name"

git config --global user.email "your.email@example.com"

* + **SSH Key：**
    - 生成 SSH Key：ssh-keygen -t rsa -b 4096 -C "your.email@example.com"
    - 将生成的 SSH Key 添加到 GitHub 账户中。

**3. 开发工具**

* **集成开发环境 (IDE)：**
  + **Visual Studio Code (VS Code)：** 推荐使用 VS Code 作为主要开发工具，其具有轻量级、跨平台、插件丰富等优点。
  + **插件：** 建议安装以下 VS Code 插件：
    - Python：提供 Python 语言支持。
    - Pylance：提供 Python 代码智能提示和静态分析。
    - GitLens：增强 Git 功能，方便查看代码提交历史和差异。
    - EditorConfig：统一代码风格。
    - 其他：根据个人需求选择安装。
* **其他工具：**
  + **命令行工具：** Windows Terminal 或 PowerShell。
  + **代码格式化工具：** Black 或 autopep8。
  + **单元测试框架：** pytest 或 unittest。

**4. 开发规范**

* **代码风格：**
  + 遵循 PEP 8 编码规范。
  + 使用 EditorConfig 统一代码风格。
  + 使用代码格式化工具 Black 或 autopep8 格式化代码。
* **版本控制：**
  + 使用 Git Flow 或 GitHub Flow 工作流。
  + 提交代码前进行代码审查。
  + 编写清晰的提交信息。
* **文档：**
  + 编写详细的项目文档，包括用户手册、开发文档、API 参考等。
  + 使用 Markdown 格式编写文档。
* **测试：**
  + 编写单元测试和集成测试，提高代码质量。
  + 使用 pytest 或 unittest 运行测试。
* **其他：**
  + 团队成员之间保持沟通，及时解决问题。
  + 定期进行代码重构，优化代码结构。

**5. 开发流程**

1. **环境搭建：**
   * 安装操作系统、Git、开发工具等。
   * 配置 Git 全局用户名和邮箱。
   * 生成并添加 SSH Key。
   * 安装项目依赖的第三方库。
2. **代码仓库：**
   * 从 GitHub 克隆项目代码。
   * 创建新的功能分支或修复分支。
3. **开发：**
   * 在自己的分支上进行开发和测试。
   * 编写符合规范的代码。
   * 提交代码到自己的分支。
4. **代码审查：**
   * 发起 Pull Request (PR) 将自己的分支合并到 develop 分支。
   * 其他团队成员进行 Code Review。
   * 根据建议修改代码。
   * PR 合并到 develop 分支。
5. **发布：**
   * 定期将 develop 分支合并到 main 分支，发布新版本。

**6. 团队协作**

* **沟通：**
  + 使用 Slack 或其他工具进行团队沟通。
  + 定期召开团队会议，讨论项目进展和问题。
* **协作：**
  + 使用 GitHub Issues 跟踪 bug、feature request 和任务。
  + 使用 GitHub Projects 或其他项目管理工具规划和跟踪项目进度。

**7. 其他**

* **持续集成：** 引入持续集成 (CI) 工具，实现代码的自动化构建和测试。
* **代码审查：** 团队成员之间进行代码审查，提高代码质量。
* **知识共享：** 鼓励团队成员分享学习资源和经验。

**总结**

本规划文档旨在统一团队的本地开发环境，提高开发效率和代码质量。请团队成员认真阅读并遵守本规范，共同打造优秀的 SAM 项目。