1. 版本管理思想

版本管理是现在软件开发思想，是为了彼此不内耗和信用协同创造出来的东西，写这几个常用版本管理软件的就是我们平时看科普文中描述的真大神（我没记他们几个名字，主要是英文名记不住，但不妨碍他们的伟大）

1. SVN分支用途和重分支

分支主要是用来隔离影响，有的使用使用库分离，有时候使用权限分离。而SVN的分支是重分支，也意味着SVN服务器上需要单独为分支开辟一份存储空间Copy当前分支的内容信息作为存本。Tag则是轻量级的，做个记录

1. SVN冲突

常规冲突图标，如果是纯代码脚本类的可以对比一下选取可用方案，而蓝图类的资产暂时无法比对，这时候需要预防或做副本选择。一般情况下用theirs（既SVN服务器版本），自己的版本还在，先copy改一下后缀名放在其他路径下，通过两次UE编辑器启动关闭对比两个版本，不允许无声覆盖别人版本，只能重写兼容。真没钱

让人恶心的树冲突是怎么来的？有人提交改名或删除文件至SVN中，提交完了之后又把这个文件复制回相同路径下。

如何避免树冲突？踏实用重命名新路径，或者改名操作一定要删除和新增操作同时提交才会被识别为重命名。

**提交前检查**

* **更新代码**：提交前先执行 svn update，确保本地是最新版本，避免冲突。
* **测试修改**：确保代码编译通过、功能正常，避免提交未测试的代码。
* **功能单一性原则**：每次提交应包含 **单一、完整的功能或修复**，避免混合不相关的修改（可通过 svn diff 检查改动范围）。

同步提交影响结果：

涉及多人协作的文件，提交时注明影响范围，必要时用群聊或口头喊 @相关成员

比如高兴地说自己完成了某功能，可以有底气的喊出来，获得鼓励

**四、示例：优秀提交信息模板（太繁杂了，大家都不用）**

plaintext

[OrderSystem] Add: 新增订单超时自动关闭功能

- 实现定时任务扫描超时未支付订单

- 新增配置项 `order.timeout` 控制超时时间（单位：分钟）

- 关联需求文档：https://xxx.com/req/456

**注意事项**

1. **保持简洁**：提交信息避免冗长，关键信息前置。
2. **中英文统一**：团队内统一使用中文为主描述，避免混用，保留专业术语原英文
3. **追溯性**：关联项目管理工具中的 Issue / 任务编号（如 Jira、Trello 编号），方便后期追溯。
4. **回滚准备**：重大修改前可先备份或记录回滚步骤。

通过规范提交行为，可提升代码审查效率、简化版本追溯，并减少协作中的沟通成本。具体规范可根据团队规模和项目需求调整，核心是确保提交内容可理解、可验证、可追溯。

在现有的条件下，如何做最简单的SVN协同？

1. 将心比心，不要让其他开发去浪费时间帮自己测代码，再追查SVN log，特别是方圆姐这类真管理层的人力资源
2. 提交不要拆分太细，也不要太粗。尽可能让资源和代码提交刚好前后两次提交，有些明显C++和蓝图依赖无法拆分确实该一次提交。
3. 为了自己不返工，想办法让美术资产给TA或美术相关部分管理，程序自己提交代码为主的文件
4. 固定频率更新代码，比如每天更新或完成某个任务完备。
5. 常备额外一个干净的SVN用于获取更新或执行提交，提交完之后。右键SVN repo-brownser看一眼是否完成想要的文件提交全了。特别是现在插件路径库和TA资产库是和标准sln路径是分开的

目前角色蓝图路径：



修改权完全由动作TA组负责，程序或其他人需要用时继承派生使用。程序不修改此路径下蓝图资源。

程序蓝图路径为：ScriptAndBlueprint\Blueprints

因已有提交带了s复数，暂时先不管，后续其他路径名没有s，类似细节参照其他路径或标准虚幻路径