钩子实际上是一个处理消息的程序段，通过系统调用，把它挂入系统。

每当特定的消息发出，在没有到达目的窗口前，钩子程序就先捕获该消息，亦即钩子函数先得到**控制权**。

这时钩子函数即可以**加工处理（改变）该消息**，也可以不作处理而继续传递该消息，还可以强制结束消息的传递。

钩子都被存储在 Git 目录下的 hooks 子目录中。 也即绝大部分项目中的 .git/hooks 。 当你用 git init 初始化一个新版本库时，Git 默认会在这个目录中放置一些示例脚本。这些脚本除了本身可以被调用外，它们还透露了被触发时所传入的参数。

客户端钩子 客户端钩子分为很多种。

下面把它们分为：提交工作流钩子、电子邮件工作流钩子和其它钩子。

提交工作流钩子 前四个钩子涉及提交的过程。

pre-commit 钩子在键入提交信息前运行。 它用于检查即将提交的快照，例如，检查是否有所遗漏，确保测试运行，以及核查代码。 如果该钩子以非零值退出，Git 将放弃此次提交，不过你可以用 git commit --no-verify 来绕过这个环节。 你可以利用该钩子，来检查代码风格是否一致（运行类似 lint 的程序）、尾随空白字符是否存在（自带的钩子就是这么做的），或新方法的文档是否适当。

prepare-commit-msg 钩子在启动提交信息编辑器之前，默认信息被创建之后运行。 它允许你编辑提交者所看到的默认信息。 该钩子接收一些选项：存有当前提交信息的文件的路径、提交类型和修补提交的提交的 SHA-1 校验。 它对一般的提交来说并没有什么用；然而对那些会自动产生默认信息的提交，如提交信息模板、合并提交、压缩提交和修订提交等非常实用。 你可以结合提交模板来使用它，动态地插入信息。

commit-msg 钩子接收一个参数，此参数即上文提到的，存有当前提交信息的临时文件的路径。 如果该钩子脚本以非零值退出，Git 将放弃提交，因此，可以用来在提交通过前验证项目状态或提交信息。 在本章的最后一节，我们将展示如何使用该钩子来核对提交信息是否遵循指定的模板。 post-commit 钩子在整个提交过程完成后运行。 它不接收任何参数，但你可以很容易地通过运行 git log -1 HEAD 来获得最后一次的提交信息。 该钩子一般用于通知之类的事情。

….

服务器端钩子

除了客户端钩子，作为系统管理员，你还可以使用若干服务器端的钩子对项目强制执行各种类型的策略。 这些钩子脚本在推送到服务器之前和之后运行。 推送到服务器前运行的钩子可以在任何时候以非零值退出，拒绝推送并给客户端返回错误消息，还可以依你所想设置足够复杂的推送策略。