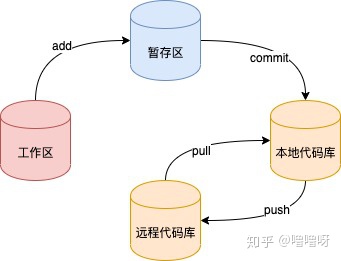
1. **General**



Git add . //添加当前目录中所有文件更改

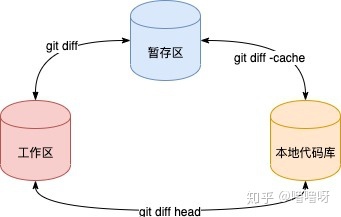
Git add filename //逐个添加文件

Git add -A // 添加当前目录中所有文件

Git commit -m “description ” //提交工作区改动到本地代码库

Git push // 提交本地代码库内容改动到远程代码库

Git pull // 下载远程代码库内容到本地代码库



**工作场景二 —— 开发进行一半，需要远端主分支的最新代码**

有些时候，你在本地开发某个功能，代码写到一半，某个同事将某些重要代码合进了**远端的主分支（如 develop 分支）**里。这些重要代码可能是可以极大提升本地开发效率，可能是加入了某些规范检查或者是跟你当前开发相关的代码 —— 总之需要你将那部分代码融入你当前的本地开发环境里。

这种情况下，我会

* **git st** 查看当前项目的状态，如果有未保存的修改，就**git add .** 和 **git ci -m "xyz"** 保存下来
* **git pull --rebase origin develop** 使用这个指令将远端的主分支以 rebase 的形式 “合进”当前分支
* **git logl** 查看当前分支下的 commit message 是否符合预期

为什么用 --rebase 呢？

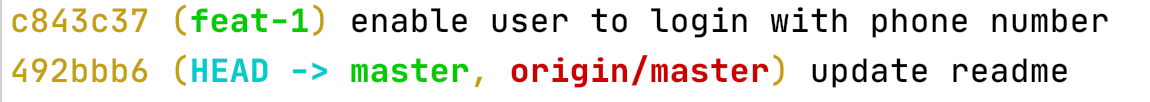
因为这么做，可以让git历史最干净、整洁 —— 所有本地开发的 commit 都会出现在远端主分支里的 commit 之后；并且可以避免额外引入一次 merge 的 commit

作者：FreewheelLee  
链接：https://zhuanlan.zhihu.com/p/250493093  
来源：知乎  
著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。

**工作场景三 —— 希望把某个分支中的某个 commit 对应的代码复制到当前分支**

有时我会创建一些实验性的分支，写一些实验性的代码，如果代码不可行，我可以直接废弃掉这个分支，返回原本的开发分支中；如果这些实验性代码可行，我会返回开发分支中，然后把实验性分支的那部分代码“复制”过来，具体操作如下：

* **git st** 查看当前项目的状态，如果有未保存的修改，就**git add .** 和 **git ci -m "xyz"** 保存下来
* 假设我们需要的是 feat-1 这个分支的某个 commit ，使用 **git logl feat-1** 查看最近这个分支的所有 commit 记录，如



记下对应的 commit 的 hash 值，即开头的7个字符乱码 如上图的 c843c37

* 使用 **git cherry-pick c843c37** 将这个 commit 对应的代码复制到当前分支

另外 git cherry-pick 有时还可以用来救场 —— 找回“丢失”的代码，前提是能找到对应的 commit hash 值。

作者：FreewheelLee  
链接：https://zhuanlan.zhihu.com/p/250493093  
来源：知乎  
著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。