使用TortoiseGit处理代码冲突

场景一：

开发人员A：有新提交

开发人员B：没有git pull —> 写新代码 —> git pull —> 提示有冲突

解决办法

stash save(把自己的代码隐藏存起来) -> 重新pull -> stash pop(把之前存起来的隐藏的代码取回来 ) -> 代码文件会显示冲突 -> 右键选择edit conficts，解决后点击编辑页面的 mark as resolved -> commit&push

场景二：

开发人员A：有新提交

开发人员B：没有git pull —> 写新代码 —> commit&push —> 提示有冲突

解决办法

重新pull -> 代码文件会显示冲突 -> 右键选择edit conficts，解决后点击编辑页面的 mark as resolved -> commit&push

分支的合并

git代码合并有两种：git Merge、git Rebase

git Merge

这种合并是将两个分支的历史合并到一起，现有的分支并不会被更改，git Merge比对双方不同的文件，缓存下来，生成一个commit，然后去push

优点: 安全，现有子分支不会被修改

缺点: 或多或少都会污染一点子分支历史，在回看项目时，会增加理解项目历史的难度

用处: 一般用于公共master主分支

git Rebase

这种合并通常称之为“衍合”，他是修改提交历史，比对双方的commit，然后找出不同的去缓存，然后在去push，修改你的commit历史。

优点: 项目历史会非常整洁

缺点: 安全性和可跟踪性很差，你将无法知晓这次合并做了那些修改

用处: 绝不要在公共的分支上使用它。一般用于，自己本身独自使用的分支

总结

这两种方式各有优点和缺点，我们要根据实际情况和需要去决定使用哪种合并方式。我的使用习惯一般是: 在我自己私有的分支，使用Rebase，保持好看的项目历史，在主master分支时使用Merge，这样安全且便于跟踪修改！