Working with Git & SourceTree

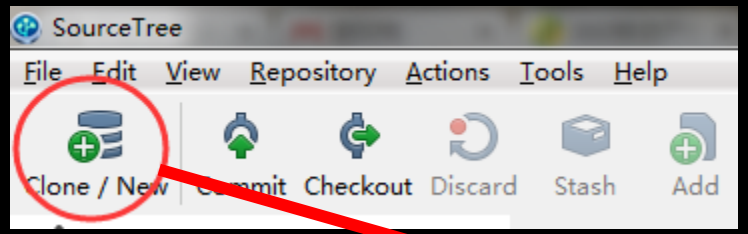
2015-05-03

V1.0

1 开始一个新项目

Clone / New

开始一个新项目（Clone / New）



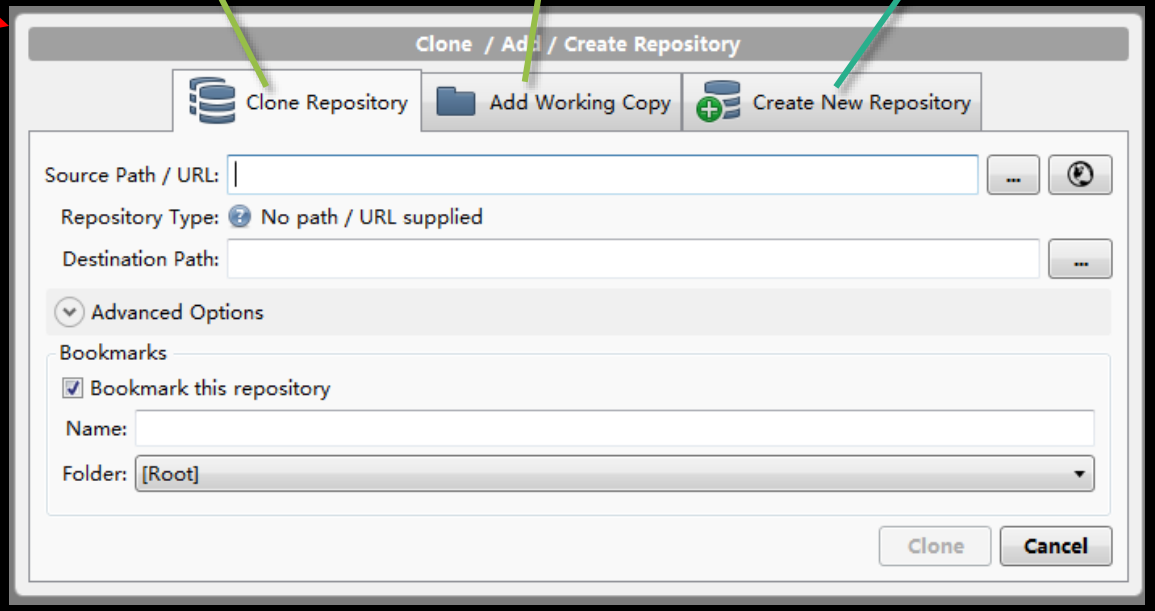
已有仓库

未有仓库，需新建

克隆现有仓库
（远程or本地均可）

导入本地仓库

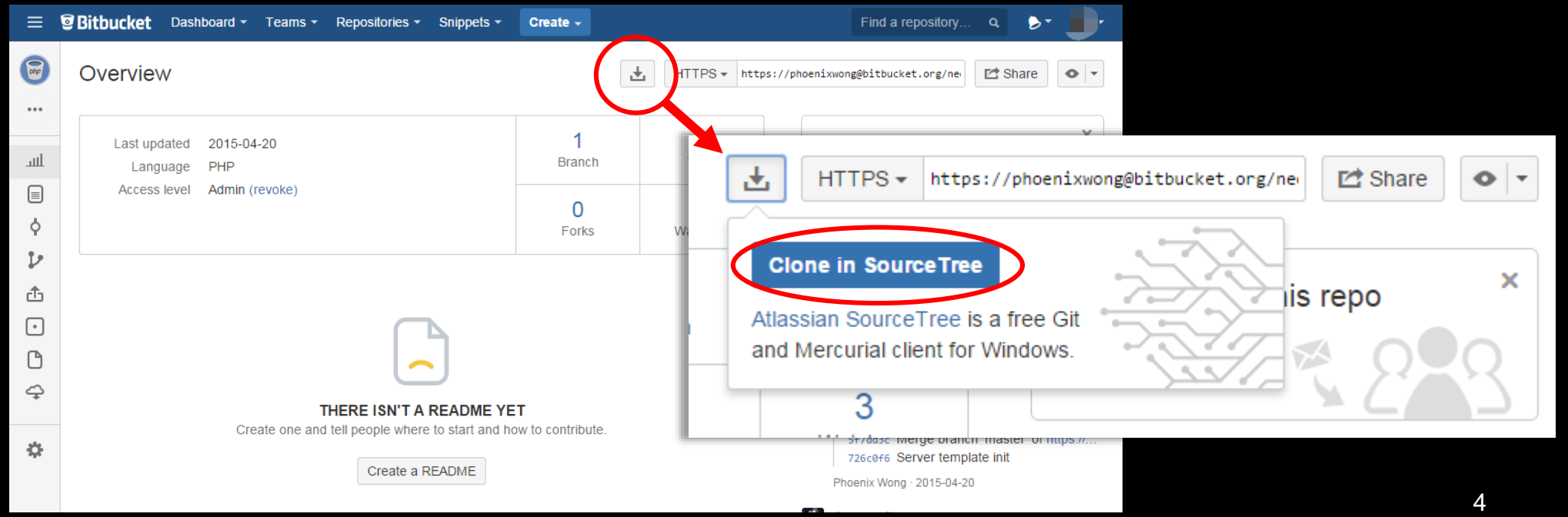
新建本地仓库



1.1 克隆现有仓库（Clone Repository）

1.1.1 从BitBucket或Stash中克隆

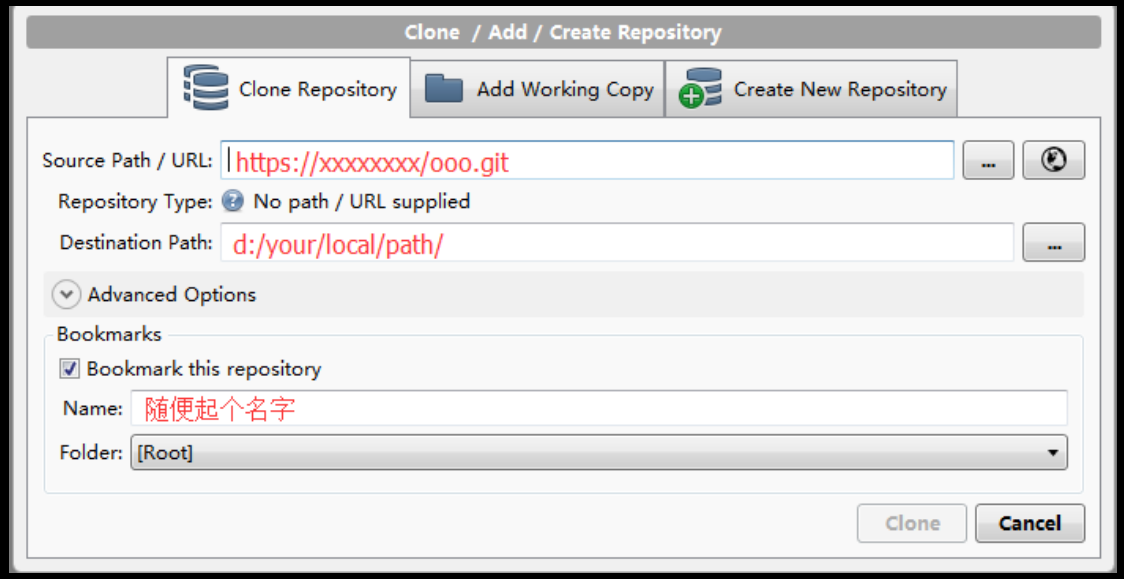
如果使用的是BitBucket或者Stash的远程仓库，在每个仓库的首页就有“一键克隆到SourceTree”的功能



1.1 克隆现有仓库（Clone Repository）

1.1.2 从其他远程仓库中克隆

其他非BitBucket和Stash的远程仓库，例如我们本地的服务器，则需要手动复制git路径（通常是https 或 ssh开头，并以 *.git结尾）。

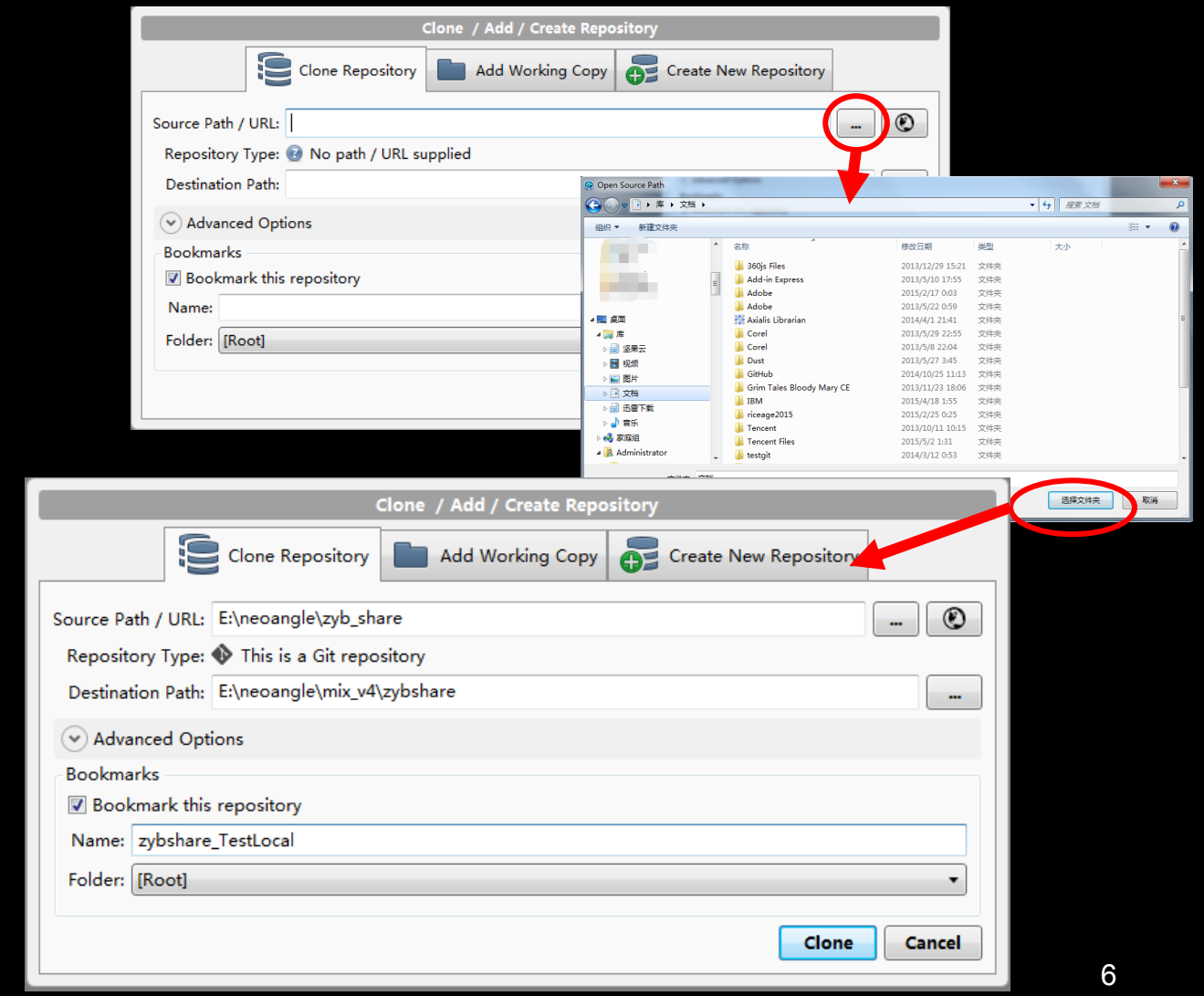


1.1 克隆现有仓库（Clone Repository）

1.1.3 克隆现有的本地仓库

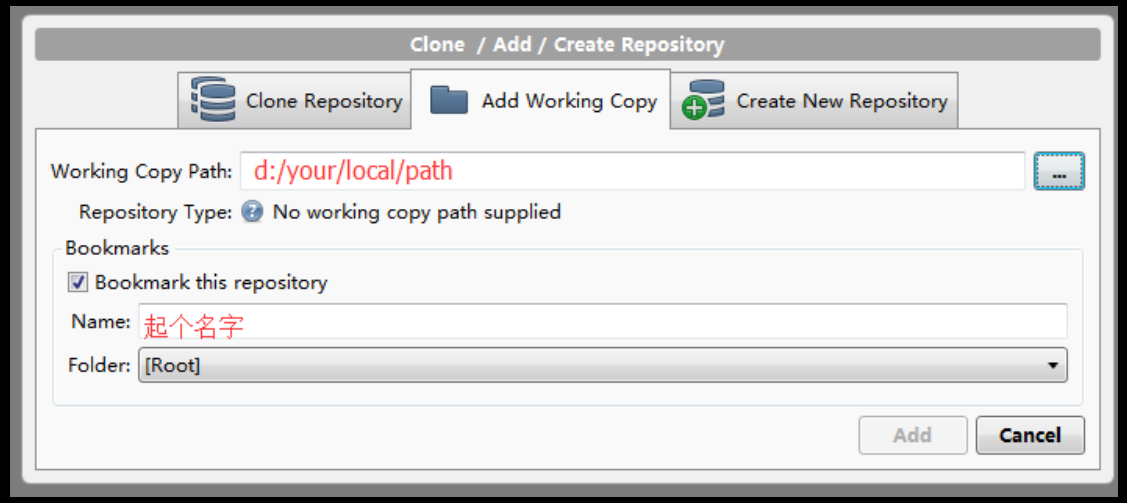
通过这个对话框还可以对本地仓库进行克隆。例如：克隆d:/zyb1项目为一个新的e:/zyb2项目。

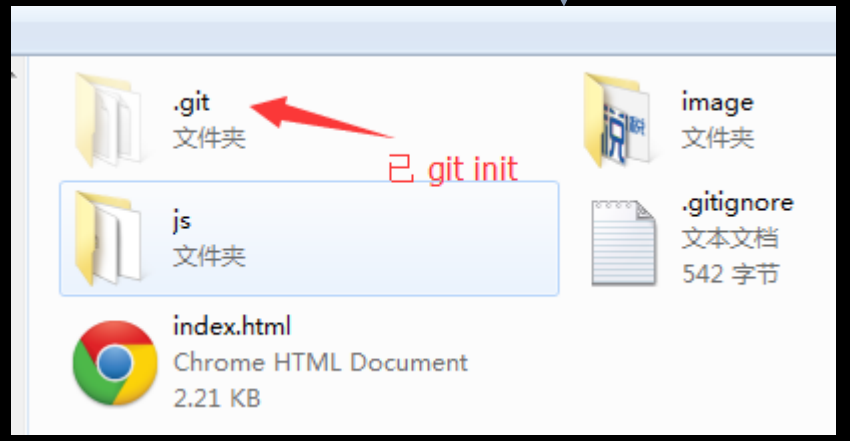
- 通过这样克隆后，zyb1就成为了zyb2的“远程仓库（origin remote）”
- 即：可以从zyb2中*Fetch*和*Pull* zyb1的更新，也可以把zyb2中的更新*Push*到zyb1中。
- 由于Git自身分支功能足够强大，一般情况下我们极少使用（也不建议使用）这个本地克隆功能。



1.2 导入本地仓库（Add Working Copy）

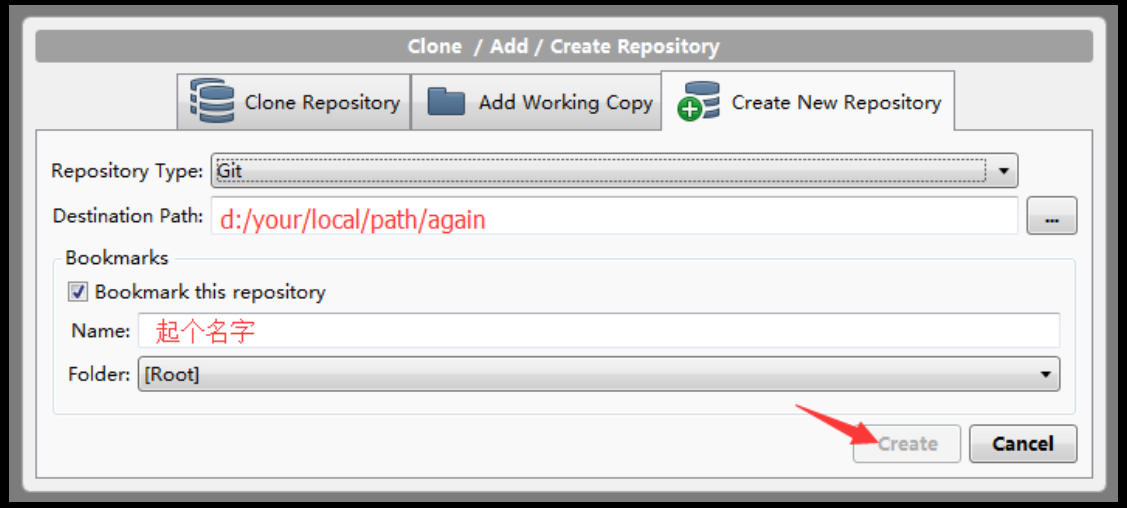
- 添加已经 `*git init*` 过的本地路径，即文件夹内已有隐藏的 .git 文件夹





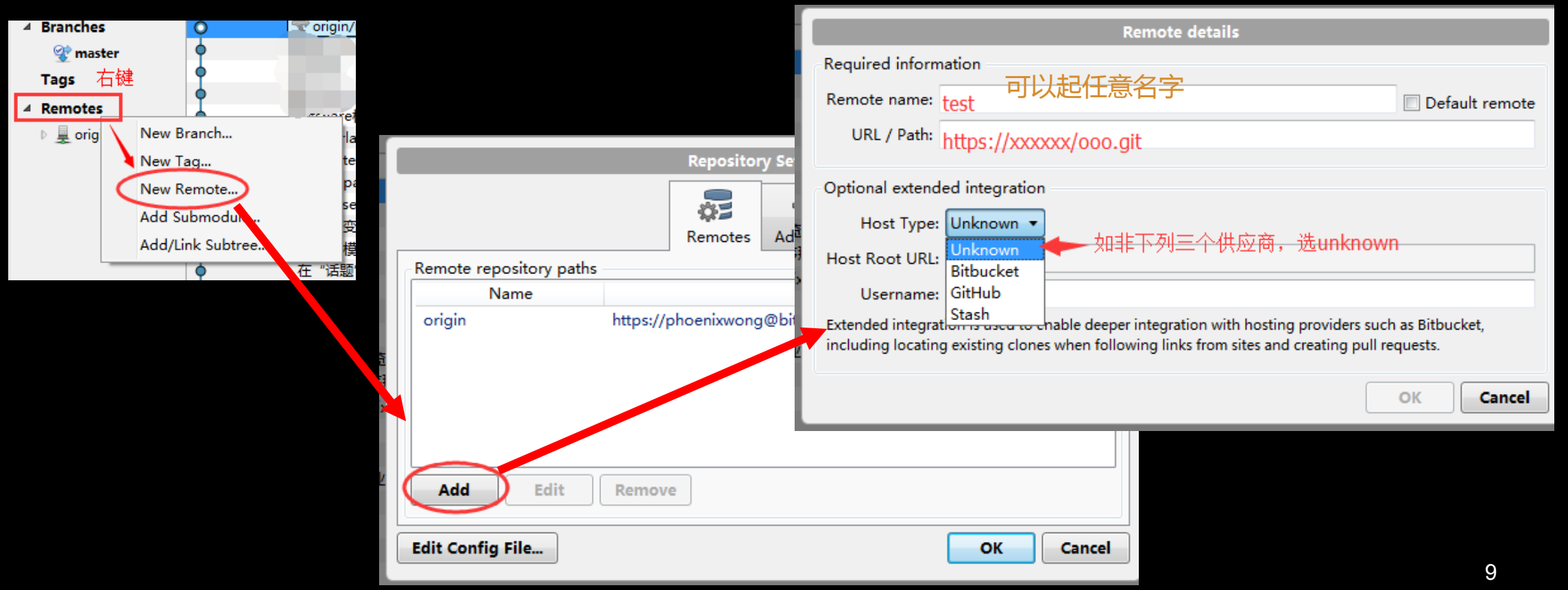
1.3 新建本地仓库（Create New Repository）

1.3.1 添加本地路径



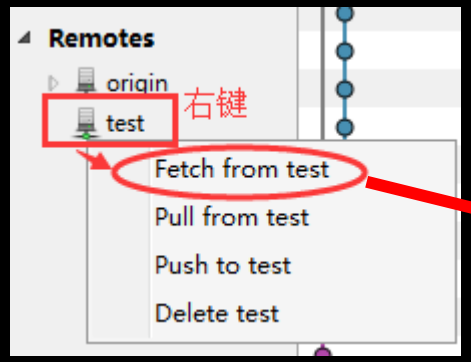
1.3 新建本地仓库（Create New Repository）

1.3.2 添加远程仓库（remote）



1.3 新建本地仓库（Create New Repository）

1.3.3 测试远程仓库是否连通



首次成功连通该远程服务器，会弹出两个框：
① 是否接受该服务器SSH key，选“接受”
② 登录验证框（如下图）

选择记住密码并Login，以后再登录同一服务器都不需要重复验证

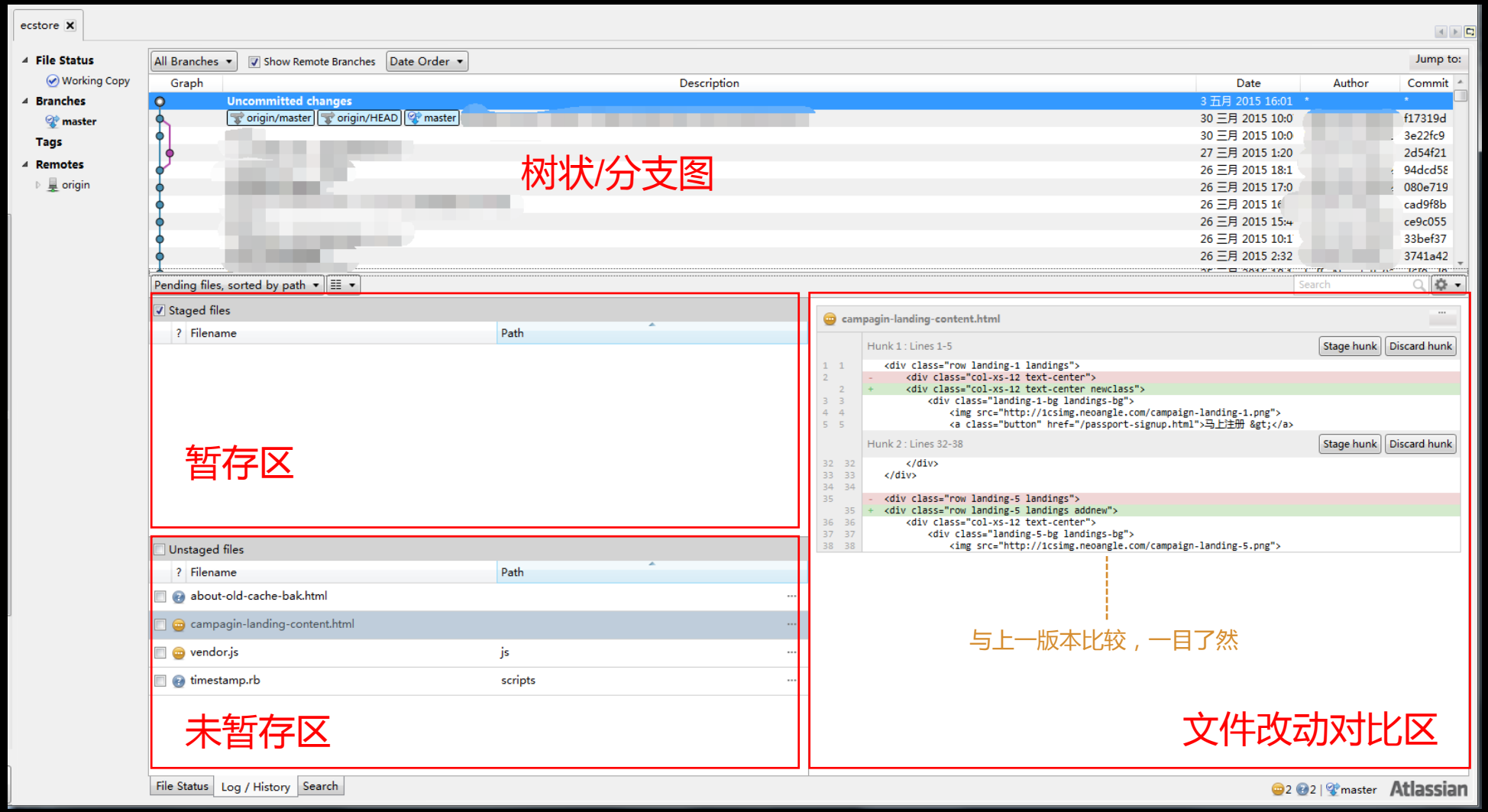
2 提交更改

Commit → Push

提交修改的三个步骤

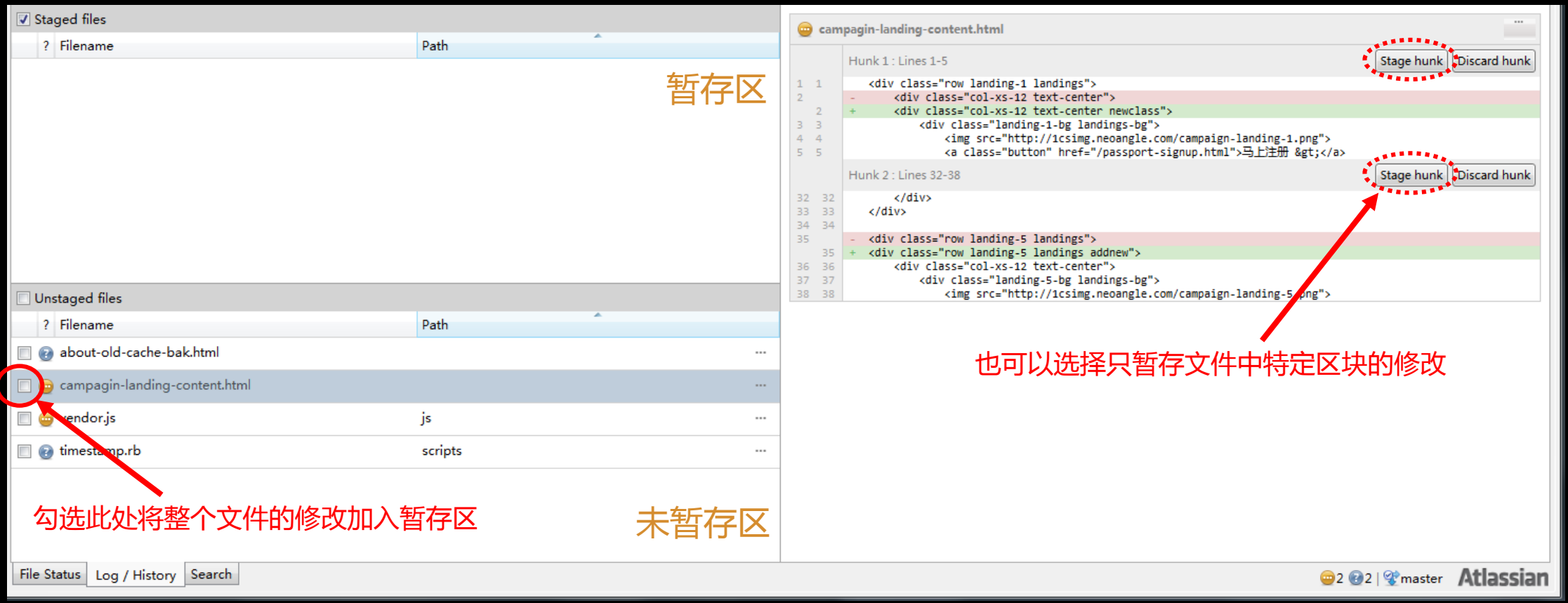
1. 将修改加入暂存区（Stage）
2. 提交修改（Commit）
3. 更新到远程服务器（Push）

附：SourceTree单个项目区块总览



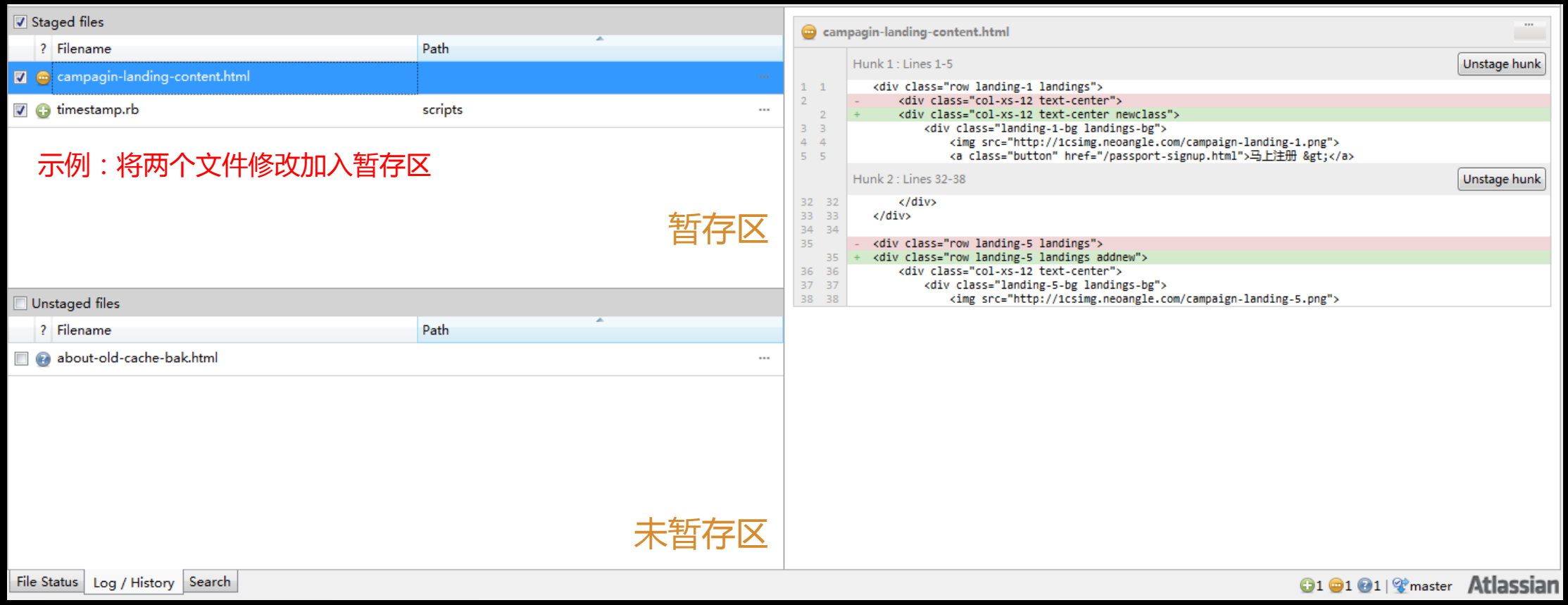
2.1 将修改加入暂存区（A）

示意图：未有任何文件加入暂存

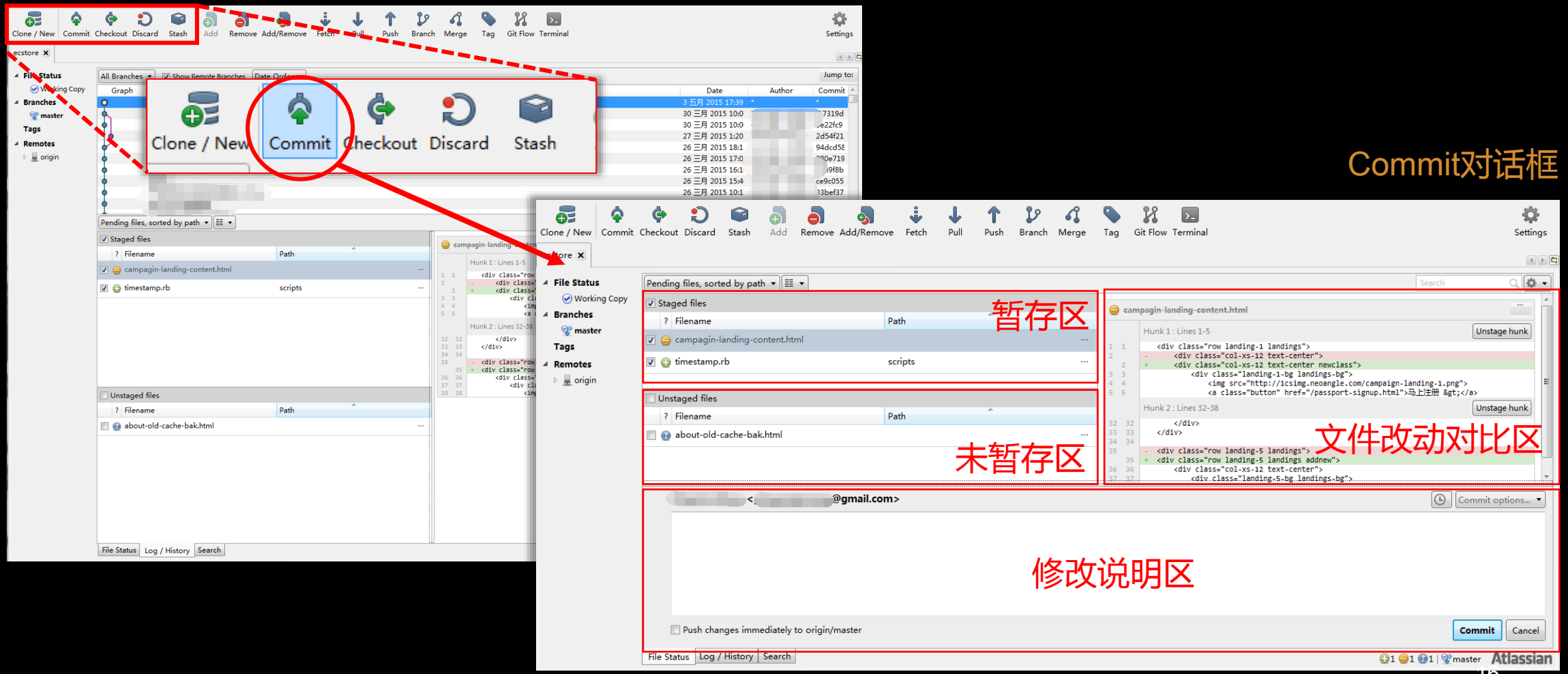


2.1 将修改加入暂存区（B）

示意图：已有两个文件加入暂存



2.2 提交修改（Commit）（A）



Commit对话框

2.2 提交修改（Commit）（B）

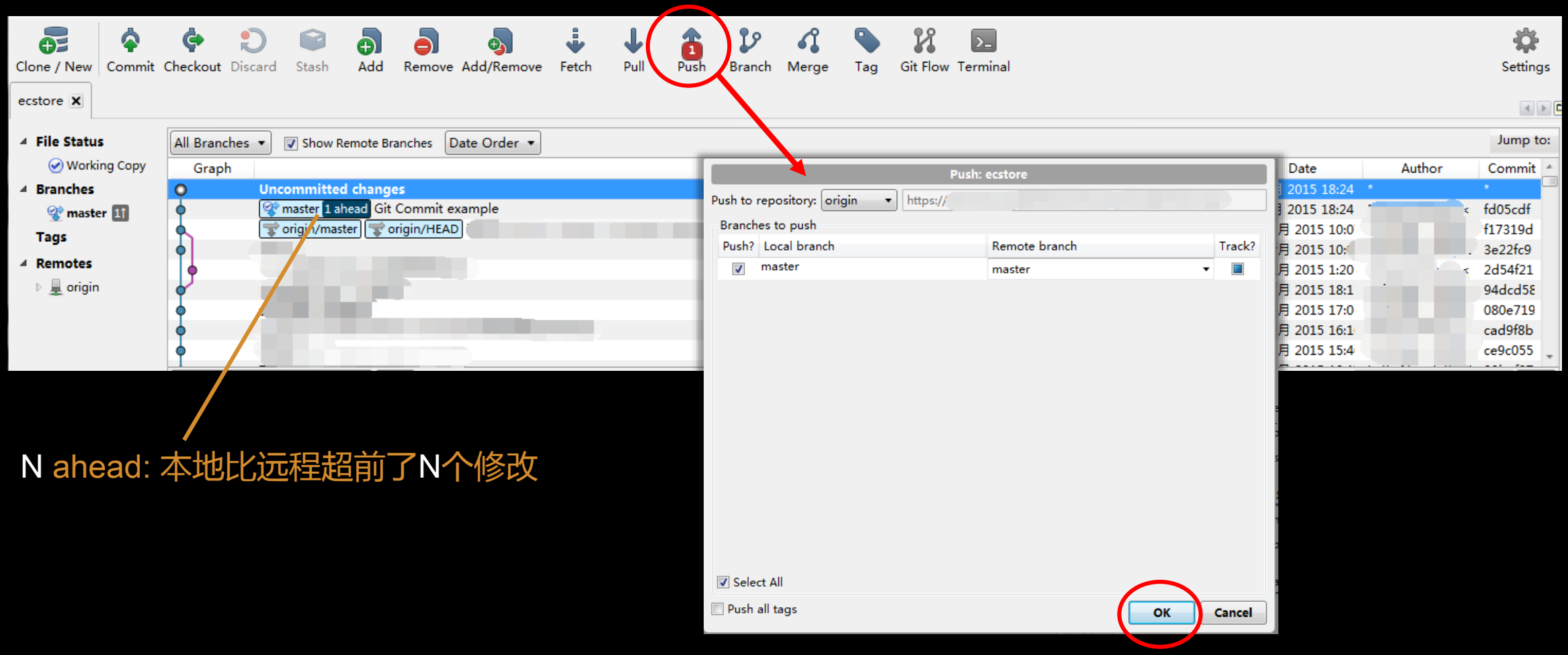
修改说明区

如果勾选了此项，则点击右侧【Commit】按钮时，会进行Commit并Push到远程服务器，合共两项操作；

如果没有勾选此项，点击【Commit】按钮时只会进行单项Commit操作，更新信息仍在本地，需要手动进行Push才能更新到远程服务器。

有人会习惯每次都Commit+Push到服务器，还有人会习惯蓄几次Commit后再Push。
纯看项目要求和个人工作习惯，没有严格的好坏或对错之分。

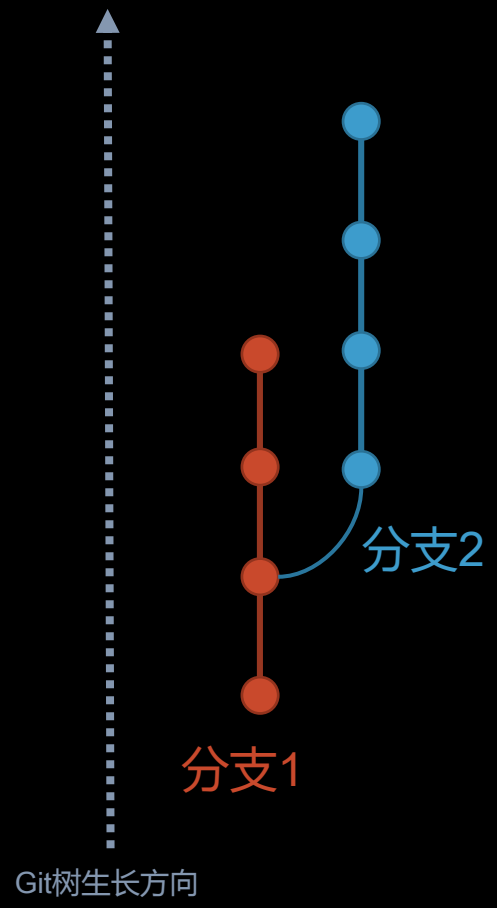
2.3 更新到远程服务器



N ahead: 本地比远程超前了N个修改

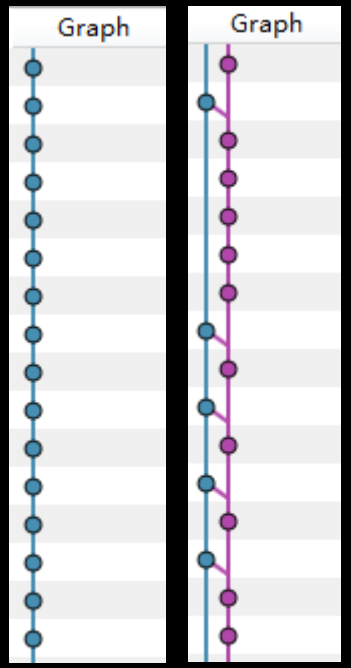
3 Git分支功能简述

3.1 Git树

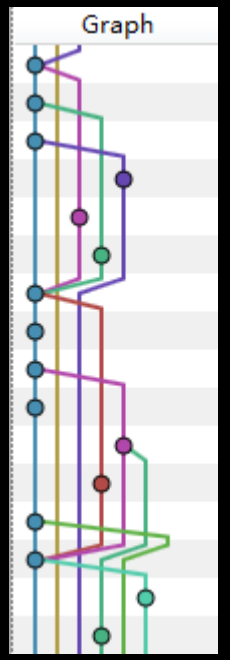


Git树一般从下往上生长，其分支的数量和走势视每个项目的复杂程度而定。

可以长得很简单



也可以开出朵花



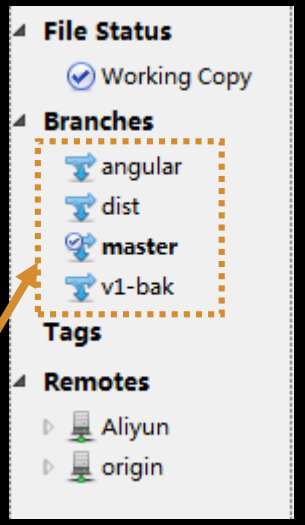
3.2 分支切换

3.2.1 本地分支切换

只要满足——

1. 分支仍存在（没有被删除）
2. 当前没有未提交的修改

你可以在任何时候切换到任意分支。



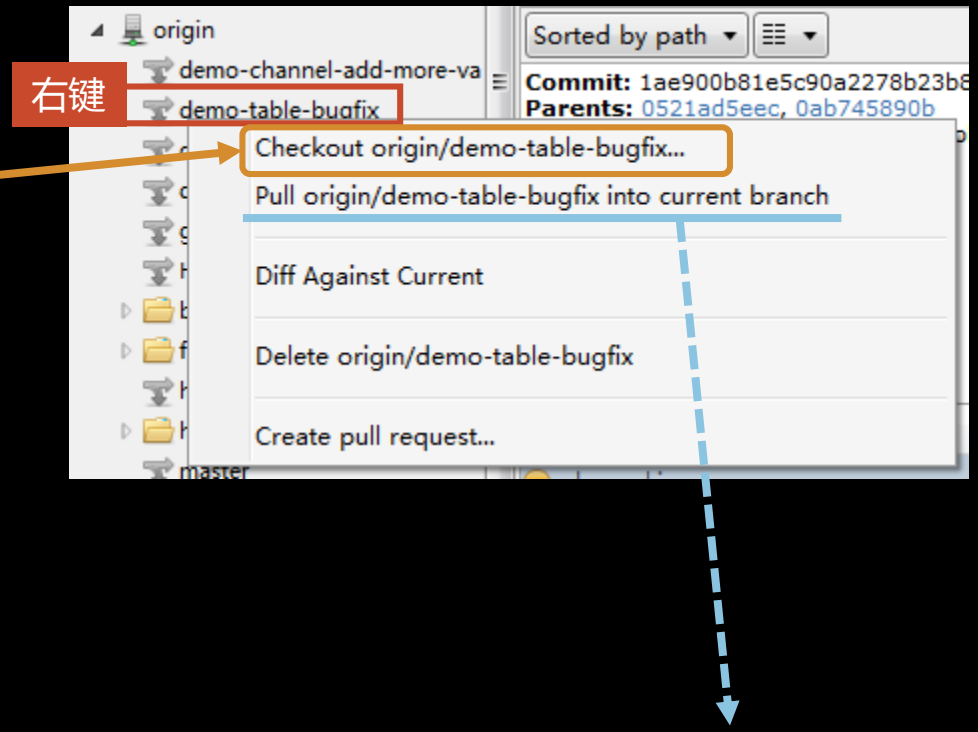
在SourceTree中，双击任意本地分支名称即可完成切换

3.2 分支切换

3.2.2 远程分支切换到本地

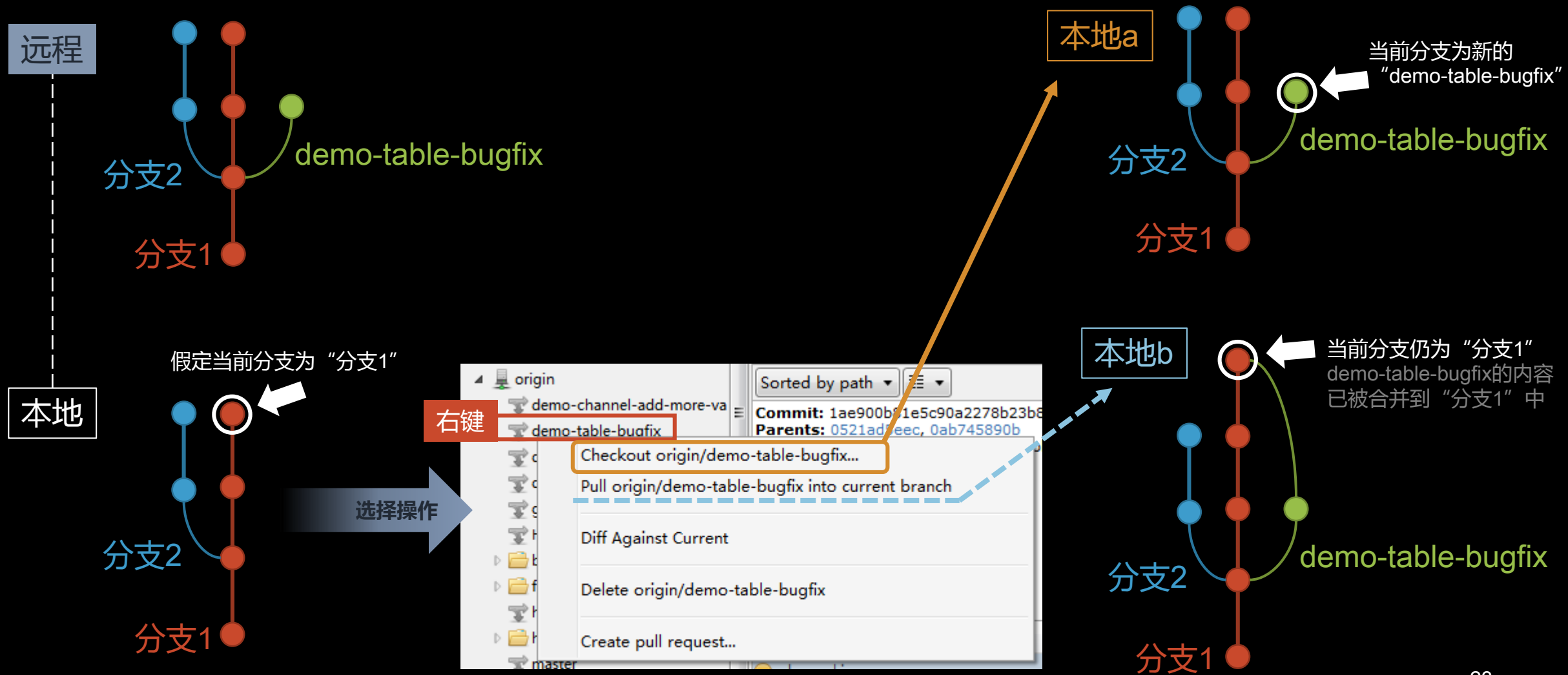
即将远程有，但本地没有的分支拉取下来，会保留分支状态。

较常用于QA复查（例如对比检查两个分支间的实际运作差异），其次多用于子版本的协作。



注意与此项进行区分——
选择此项会将远程的分支上的修改直接合并（merge）到当前的本地分支中

附：Checkout与Pull远程分支的区别



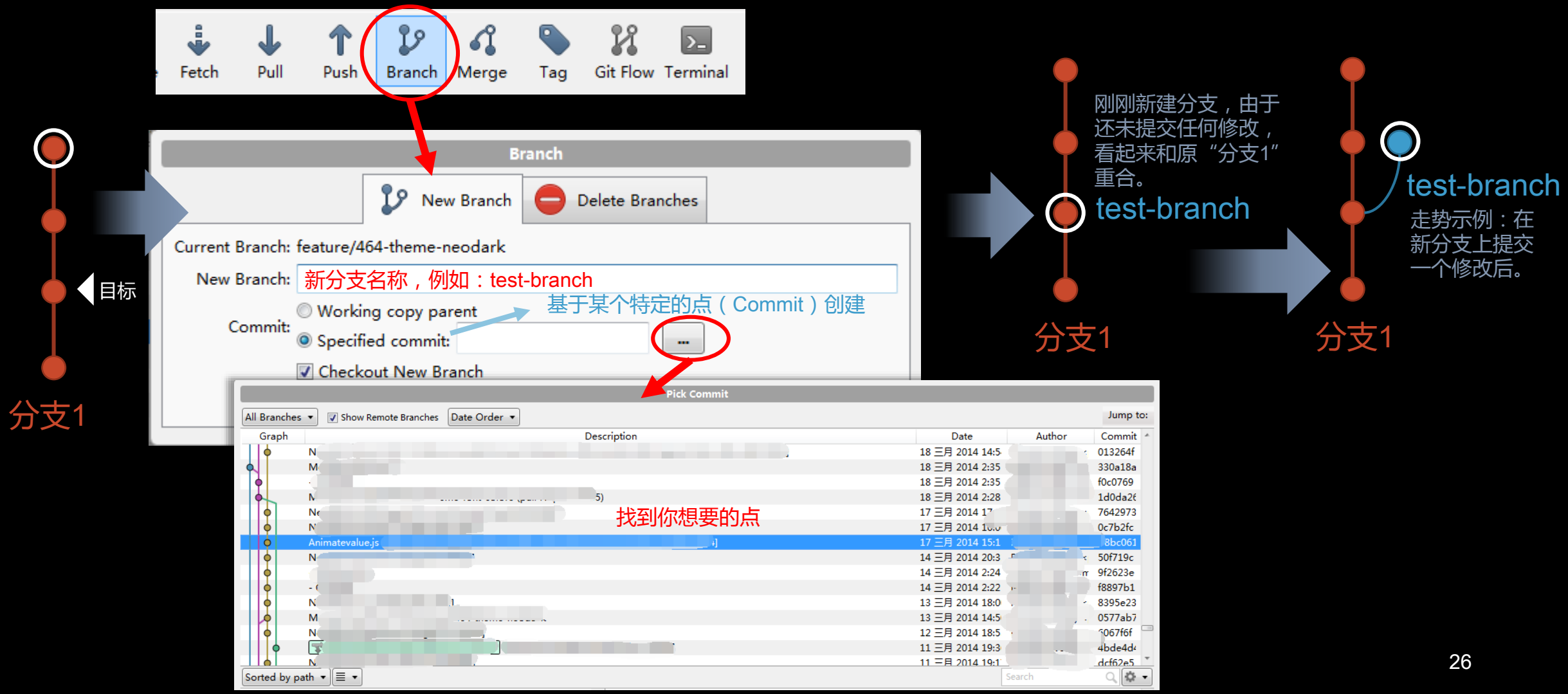
3.3 新建分支

你可以在任何时候，基于任何点新建一个分支，主要有两种方式：

1. 以当前的本地分支为基准，新建一个分支；
2. 基于某个特定的修改点（commit）新建一个分支。

3.3.1 基于当前的本地分支创建新分支

3.3.2 基于某个特定的修改点创建新分支



4 常用方法借鉴

下面将列出一些常用的分支命名和工作流程示例，仅供参考。实践时仍需依照不同的项目复杂性而择优选择。

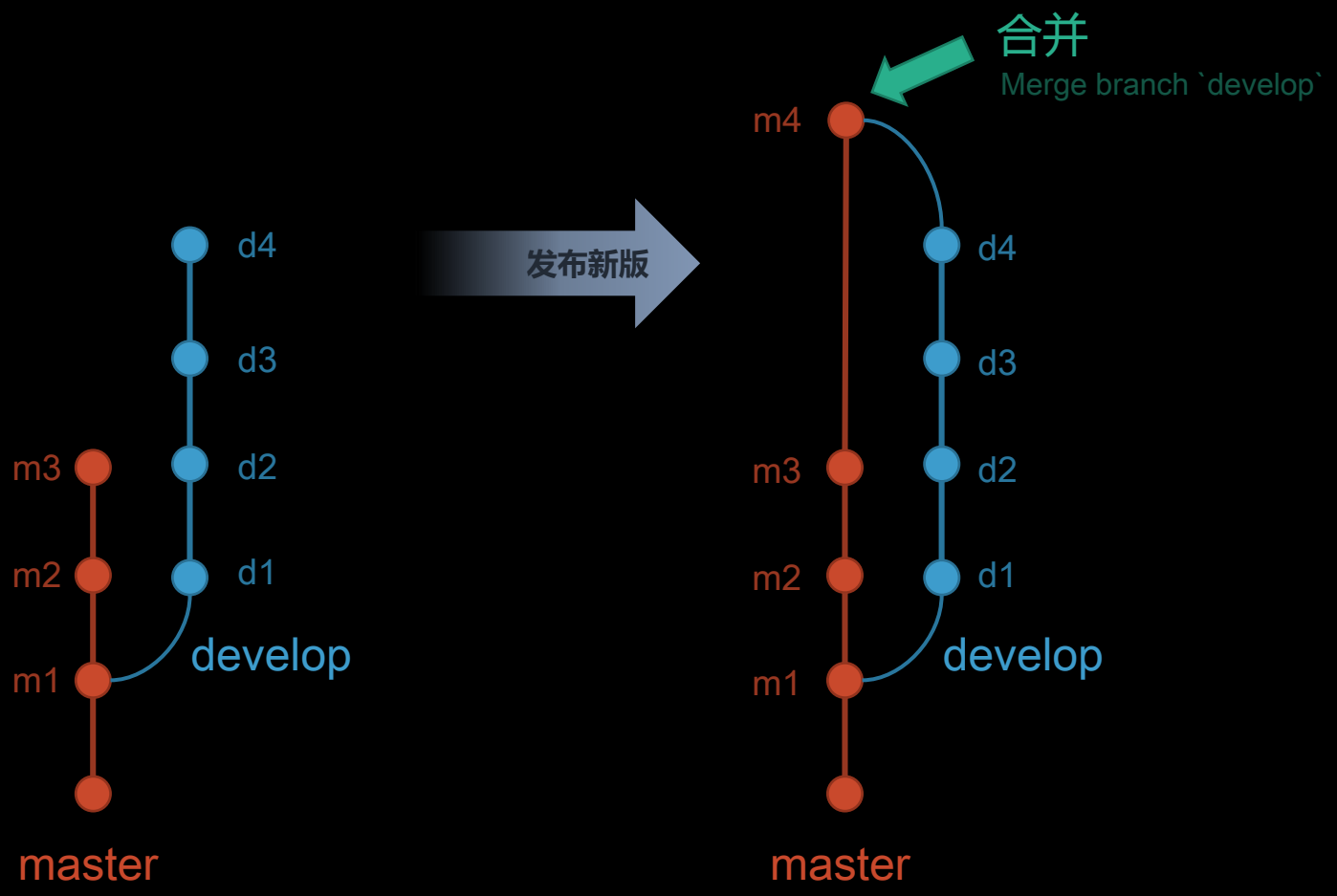
4.1 常用的分支命名和管理

问：分支可以用中文命名吗？

- 答：虽然经本地测试是可以用中文的，但为了长远考虑（不能100%确定以后每台远程服务器都支持中文分支），建议大家还是统一使用英文字符和数字命名（拼音也是可以接受的哟）

4.1.1 基础分支

- 较大型的项目一般会使用两个以上分支，其中最基础的两个分支是：
  - master：常用于连接生产环境（PROD）
  - develop：常用于连接测试环境（QA）
- 通常，develop会跑得比master快很多。
- 一般到下一个版本发布时（例如从v2.2升到v2.3），才会将develop合并到master中。

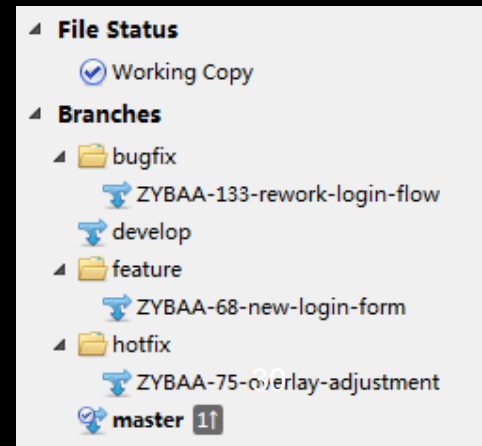


4.1.2 特色分支（Feature Branches）

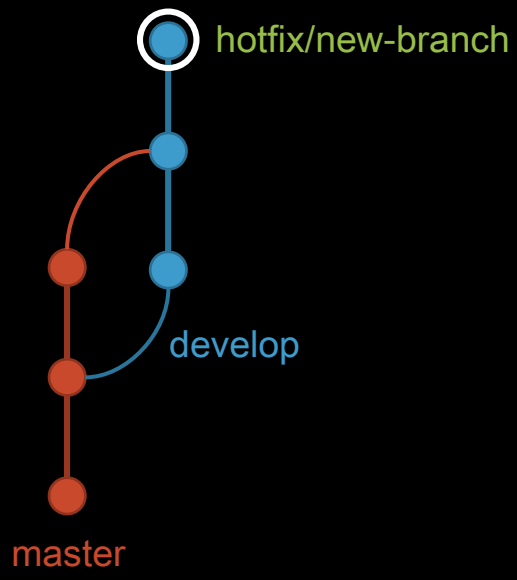
- 虽然，理论上我们可以给分支起任何名字。但为了利于长期项目协作，人们对分支的命名有一些约定俗成的规则。配合JIRA的使用，分支命名大致可分为三类：
  1. 基于测试环境的小修改，以hotfix为前缀
     - hotfix/ZYBAA-75-overlay-adjustment
  2. 新功能的开发，以feature为前缀
     - feature/ZYBAA-68-new-login-form
  3. 基于生产环境的修改，或较大的bug修改，以bugfix为前缀
     - bugfix/ZYBAA-133-rework-login-flow

* 其中，ZYBAA-nnn为JIRA中对应任务的编号

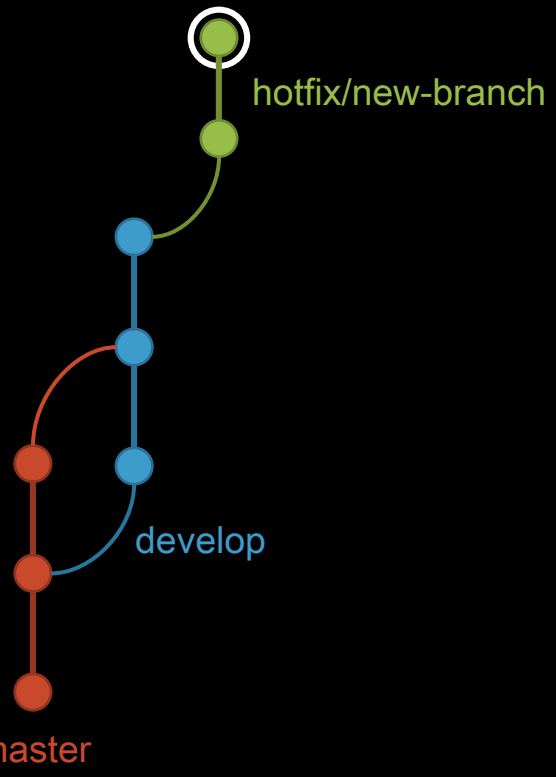
在SourceTree中的效果



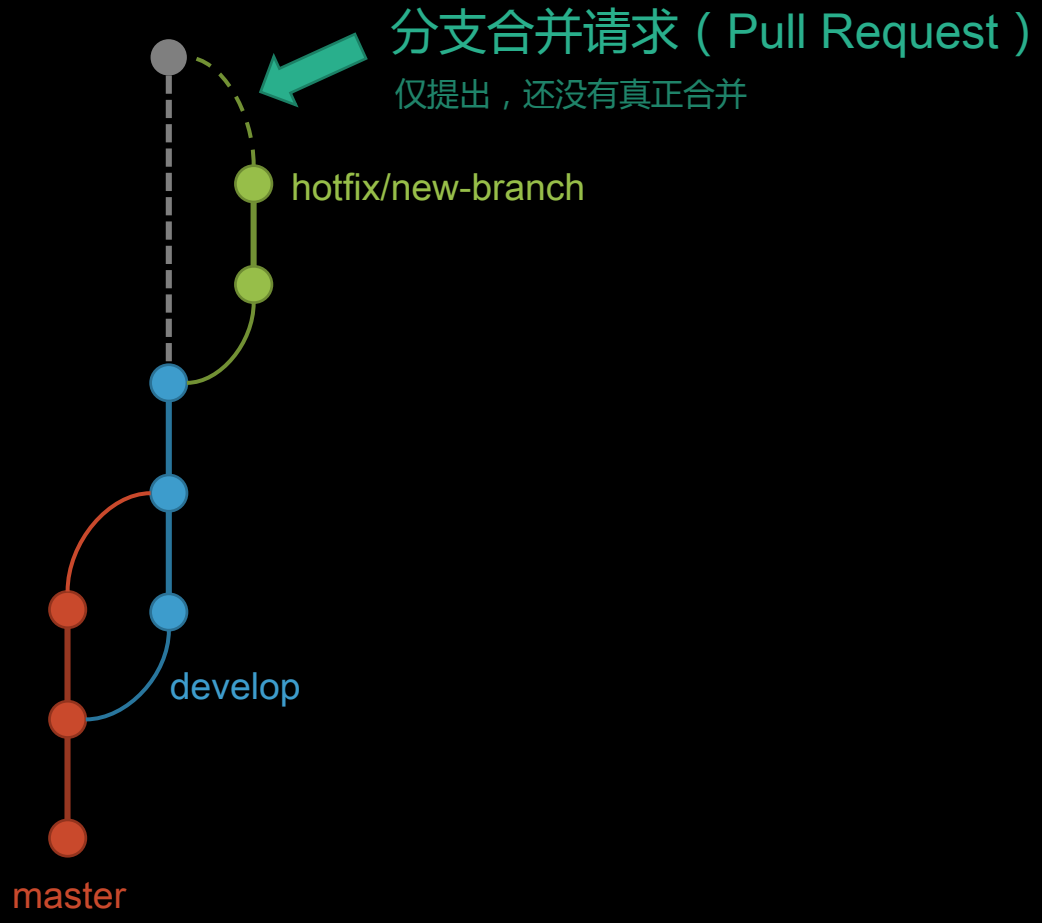
4.2 常见工作流程（A）



1 总是基于测试环境创建特色分支

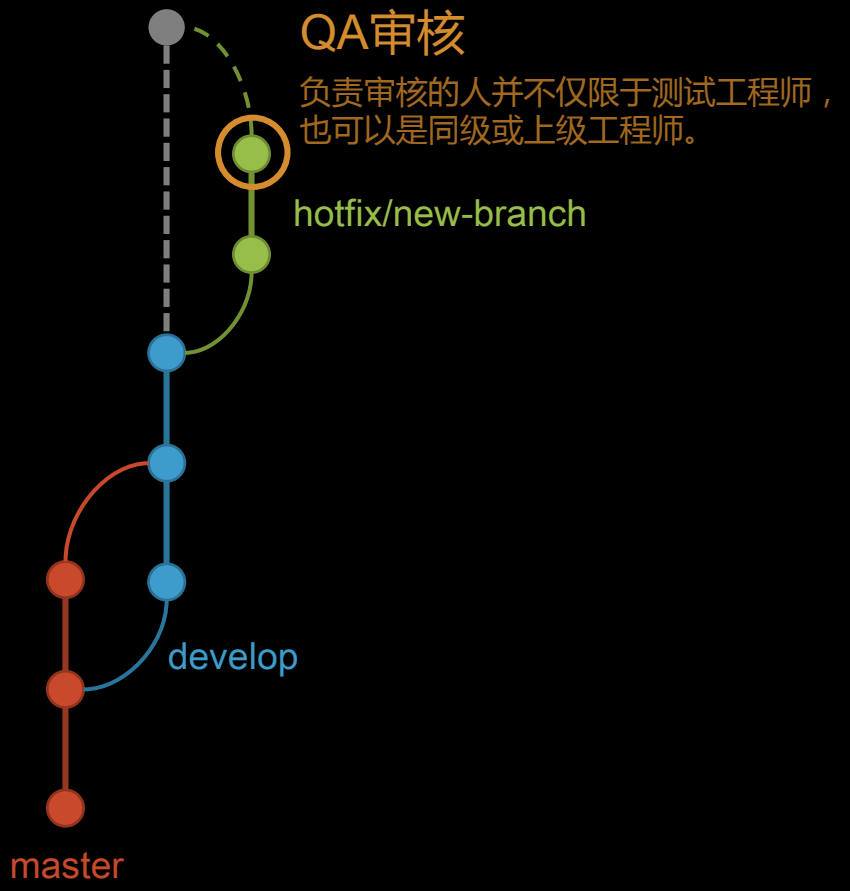


2 在特色分支上完成该修改任务

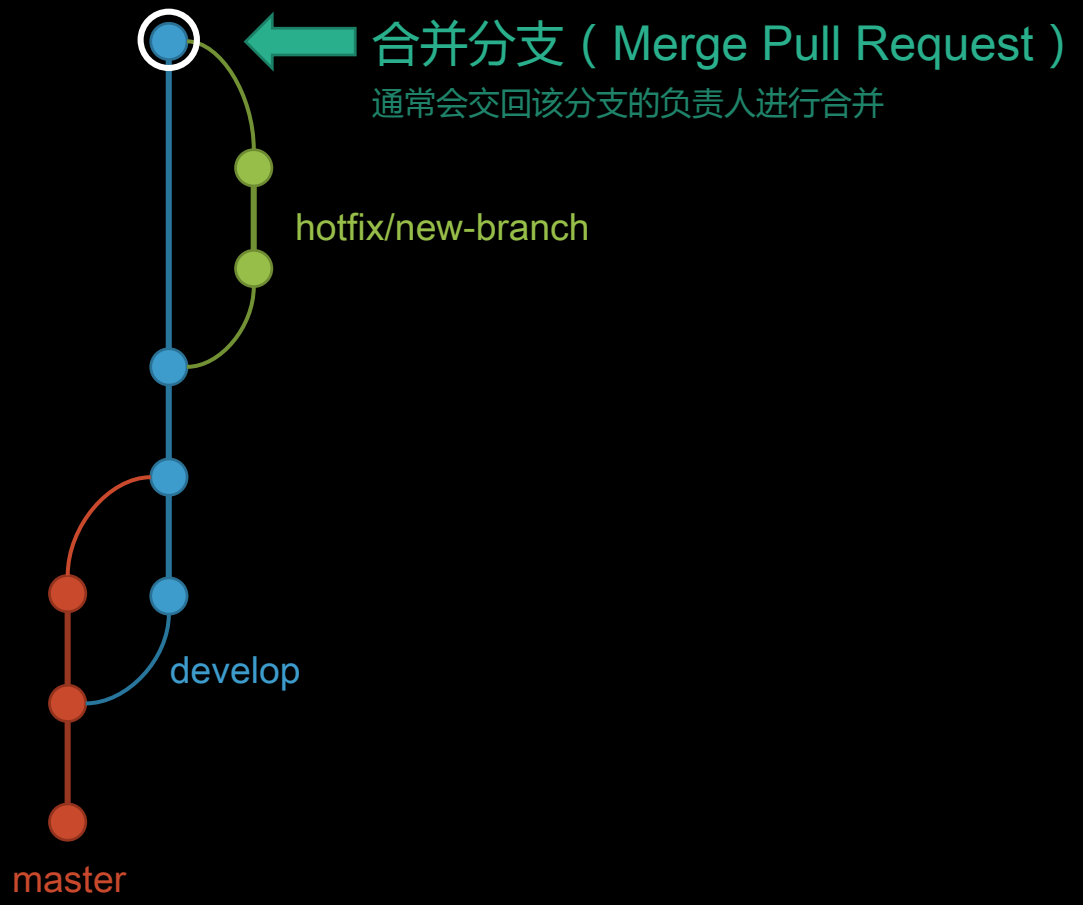


3 提出一个分支合并请求（Pull Request）

4.2 常见工作流程（B）



4 相关QA人员审核修改，确定是否允许此请求。



5 请求被通过后，相关人员可以正式将特色分支合并到测试环境

4.3 这么复杂的流程为了啥？

- 继续重申一下——上述的工作流程仅供示例。该流程较适用于大型的、多人协作的系统开发。具体优点要到实践过程中才能慢慢体会。
- 对于轻量的、只有1-2人维护的项目，使用2个左右的分支（甚至只使用一个master分支）也是可以的。

5 冲突！

多人协作难免有代码冲突，如何破？

5.1 避免冲突

- 避免代码冲突要从养成良好习惯做起：
  - 善用特色分支，基本能做到零冲突
    - 针对每个JIRA任务都新建一个特色分支
    - 每完成一个任务，提交一个该特色分支→develop的合并请求
    - （可选）等待QA审核通过后再合并该分支
  - 如果多人一起协作同一分支（不建议，否则和用SVN没太大区别了），则要经常及时将服务器的最新版本拉到本地，自己也要及时推送更新。
  - 每天下班前将自己今天的修改全部推送到远程。

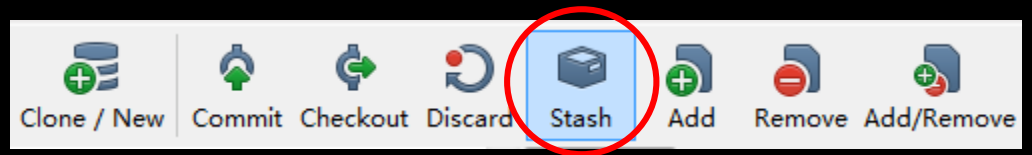
5.2 解决冲突

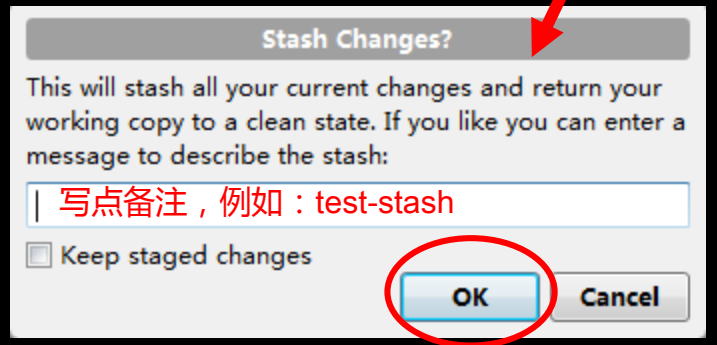
万一真的不可避免地出现冲突了，解决方法大致分三步：

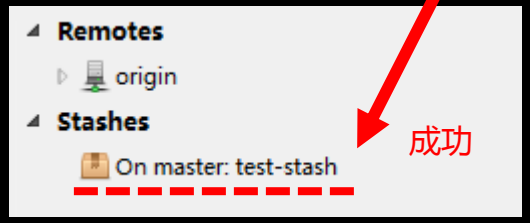
1. 将本地环境还原到未修改状态（找地方存起你的修改）
   - 方法一：隐藏修改（Stash）
   - 方法二：创建补丁（Patch）
   - 方法三：手动将相关文件copy到别的目录下
2. 将远程的最新版本拉取到本地
3. 还原你的修改

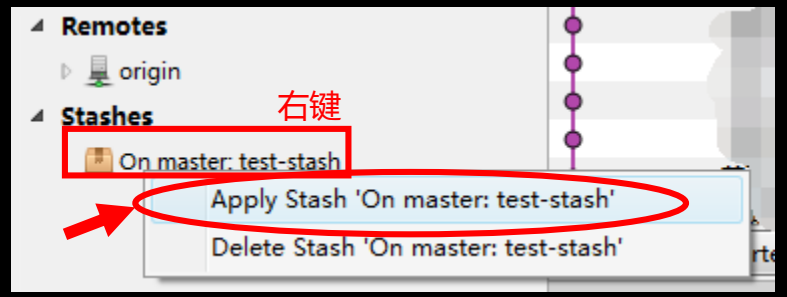
5.2.1 隐藏修改（Stash）法

存起本地修改 ⋯⋯▸ 拉取远程更新 ⋯⋯▸ 还原本地修改



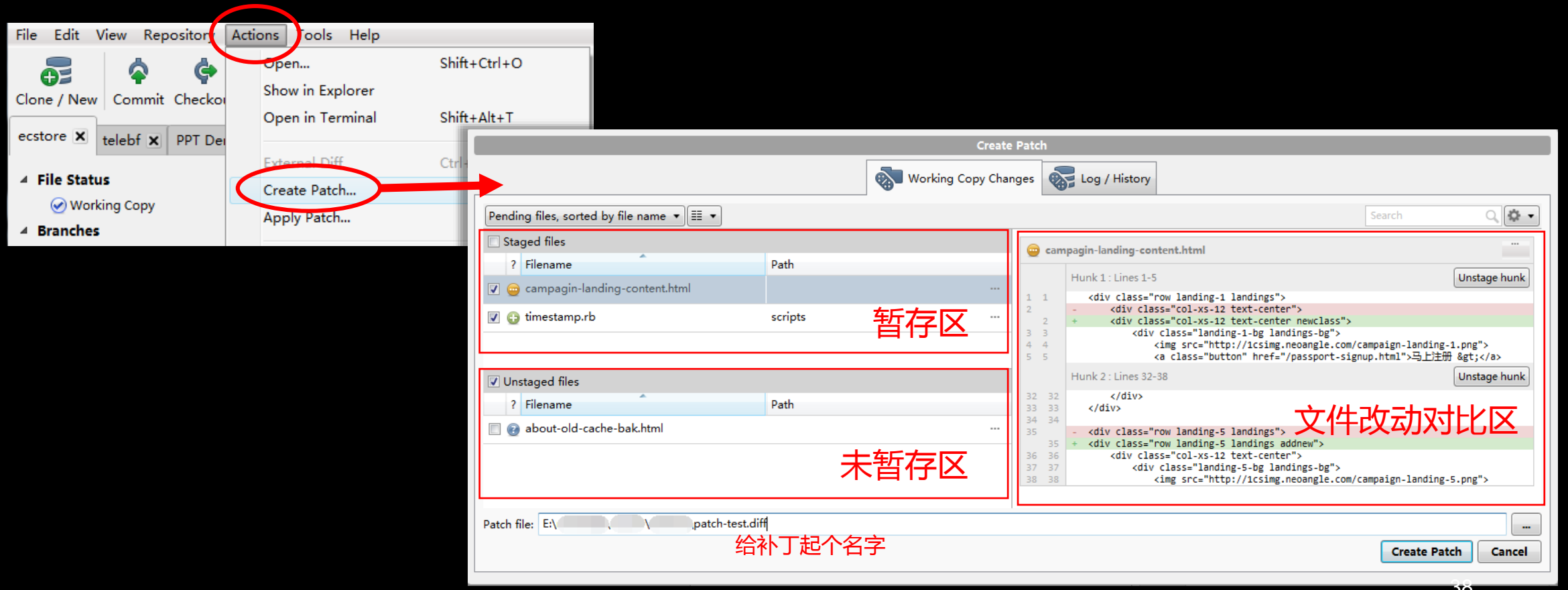






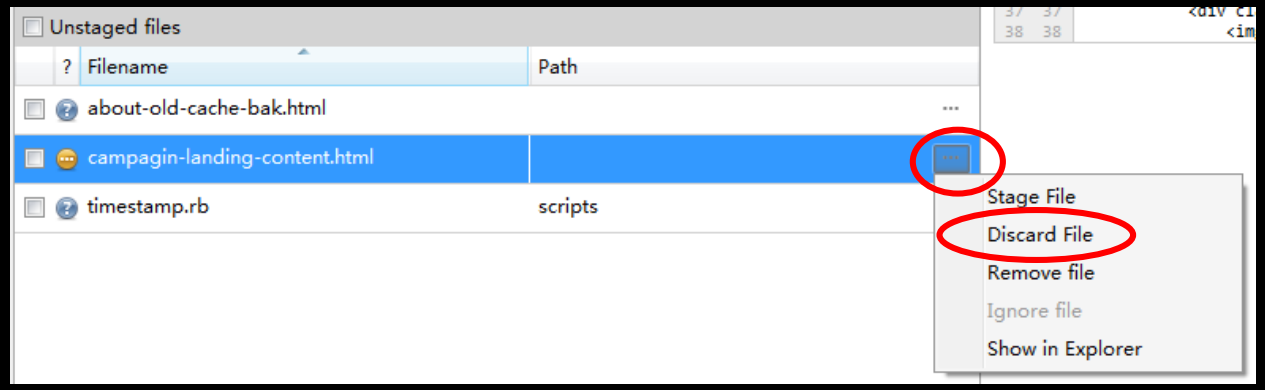
5.2.2 创建补丁法（Patch）（A）

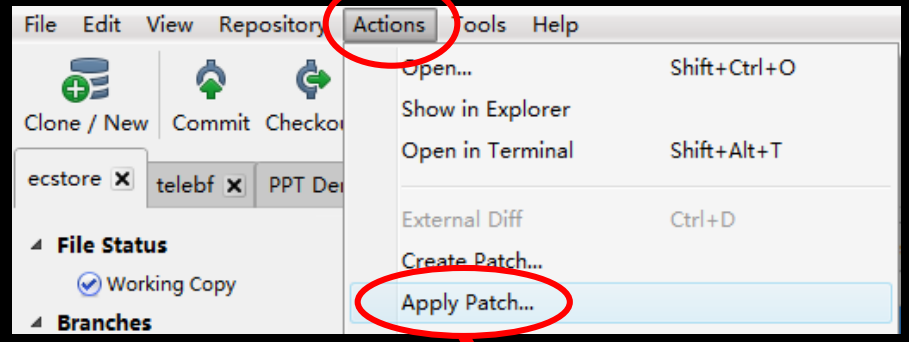
存起本地修改 ····· 下一页



5.2.2 创建补丁法（Patch）（B）

续上页 → 放弃文件更改 → 拉取远程更新 → 还原本地修改





找回原来的补丁文件，
把补丁打回去

5.2.3 手动copy法

- Heh, good luck！ :)
  - 此处省略5000字

谢谢

未完待续（？）