


[Git] Git整理(五) git cherry-pick的使用

2018年07月13日 23:49:16 FightFightFight 阅读数：22495

 版权声明：本文为博主原创文章，未经博主允许不得转载。 <https://blog.csdn.net/FightFightFight/article/details/81039050>

概述

`git cherry-pick` 可以理解为“挑拣”提交，它会获取某一个分支的单笔提交，并作为一个新的提交引入到你当前分支上。当我们需要在本地合入其他分支的提交时，如果我们不想对整个分支进行合并，而是只想将某一次提交合入到本地当前分支上，那么就要使用 `git cherry-pick` 了。

用法

```
1  git cherry-pick [<options>] <commit-ish>...
2
3  常用options:
4      --quit                退出当前的chery-pick序列
5      --continue            继续当前的chery-pick序列
6      --abort               取消当前的chery-pick序列，恢复当前分支
7      -n, --no-commit       不自动提交
8      -e, --edit             编辑提交信息
```

git cherry-pick commitid

在本地仓库中，有两个分支:branch1和branch2，我们先来查看各个分支的提交：

```
1  # 切换到branch2分支
2  $ git checkout branch2
3  Switched to branch 'branch2'
4  $
5  $
6  # 查看最近三次提交
7  $ git log --oneline -3
8  23d9422 [Description]:branch2 commit 3
9  2555c6e [Description]:branch2 commit 2
10 b82ba0f [Description]:branch2 commit 1
11 # 切换到branch1分支
12 $ git checkout branch1
13
```

```
14 Switched to branch 'branch1'
15 # 查看最近三次提交
16 $ git log --oneline -3
17 20fe2f9 commit second
18 c51adbe commit first
19 ae2bd14 commit 3th
```

现在，我想要将branch2分支上的第一次提交内容合入到branch1分支上，则可以使用 `git cherry-pick` 命令：

```
1 $ git cherry-pick 2555c6e
2 error: could not apply 2555c6e... [Description]:branch2 commit 2
3 hint: after resolving the conflicts, mark the corrected paths
4 hint: with 'git add <paths>' or 'git rm <paths>'
5 hint: and commit the result with 'git commit'
6
```

当 `cherry-pick` 时，没有成功自动提交，这说明存在冲突，因此首先需要解决冲突,解决冲突后需要 `git commit` 手动进行提交：

```
1 $ git commit
2 [branch1 790f431] [Description]:branch2 commit 2
3 Date: Fri Jul 13 18:36:44 2018 +0800
4 1 file changed, 1 insertion(+)
5 create mode 100644 only-for-branch2.txt
6
```

或者 `git add .` 后直接使用 `git cherry-pick --continue` 继续。
现在查看提交信息：

```
1 $ git log --oneline -3
2 790f431 [Description]:branch2 commit 2
3 20fe2f9 commit second
4 c51adbe commit first
5
```

branch2分支上的第二次提交成功合入到了branch1分支上。

以上就是 `git cherry-pick` 的基本用法，如果没有出现冲突，该命令将自动提交。

git cherry-pick -n

如果不想 `git cherry-pick` 自动进行提交，则加参数 `-n` 即可。比如将branch2分支上的第三次提交内容合入到branch1分支上：

```
1 $ git cherry-pick 23d9422
2 [branch1 2c67715] [Description]:branch2 commit 3
3 Date: Fri Jul 13 18:37:05 2018 +0800
4 1 file changed, 1 insertion(+)
5 $
6
```

查看提交log,它自动合入了branch1分支：

```
1 $ git log --oneline -3
2 2c67715 [Description]:branch2 commit 3
3 f8bc5db [Description]:branch2 commit 2
4 20fe2f9 commit second
5
```

如果不想进行自动合入，则使用 `git cherry-pick -n`：

```
1 # 回退上次提交，再此进行cherry-pick
2 $ git reset --hard HEAD~
3 HEAD is now at f8bc5db [Description]:branch2 commit 2
4 $ git cherry-pick -n 23d9422
5 $ git status
6 On branch branch1
7 Changes to be committed:
8   (use "git reset HEAD <file>..." to unstage)
9
10      modified:   only-for-branch2.txt
11
12 $
```

这时通过 `git status` 查看，发现已将branch2的提交获取但是没有合入。

git cherry-pick -e

如果想要在 `cherry-pick` 后重新编辑提交信息，则使用 `git cherry-pick -e` 命令，比如我们还是要将branch2分支上的第三次提交内容合入到branch1分支上，但是需要修改提交信息：

```
1 $ git cherry-pick -e 23d9422
2
3 1 [Description]:branch2 commit 3
4 2 #
5 3 # It looks like you may be committing a cherry-pick.
6 4 # If this is not correct, please remove the file
7 5 # .git/CHERRY_PICK_HEAD
8 6 # and try again.
9
```

git cherry-pick --continue, --abort , --quit

当使用 `git cherry-pick` 发生冲突后,将会出现如下信息：

```
1 $ git cherry-pick 23d9422
2 error: could not apply 23d9422... [Description]:branch2 commit 3
3 hint: after resolving the conflicts, mark the corrected paths
4 hint: with 'git add <paths>' or 'git rm <paths>'
5 hint: and commit the result with 'git commit'
```

这时如果要继续cherry-pick，则首先需要解决冲突，通过 `git add .` 将文件标记为已解决，然后可以使用 `git cherry-pick --continue` 命令，继续进行 `cherry-pick` 操作。

如果要中断这次 `cherry-pick`,则使用 `git cherry-pick --quit`，这种情况下当前分支中未冲突的内容状态将为 `modified`，

如果要取消这次 `cherry-pick`,则使用 `git cherry-pick --abort`，这种情况下当前分支恢复到 `cherry-pick` 前的状态，没有改变。

git cherry-pick < branchname >

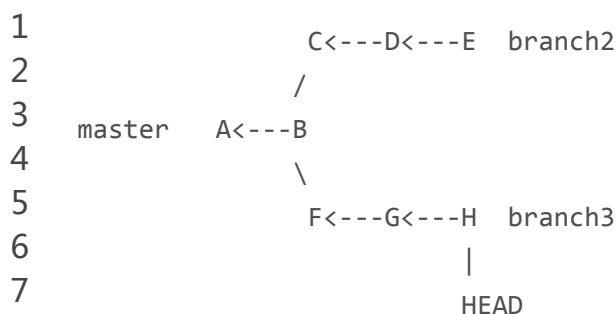
如果在 `git cherry-pick` 后加一个分支名，则表示将该分支顶端提交进cherry-pick，如：

```
1 $ git cherry-pick master
```

git cherry-pick ..< branchname >

git cherry-pick ^HEAD < branchname >

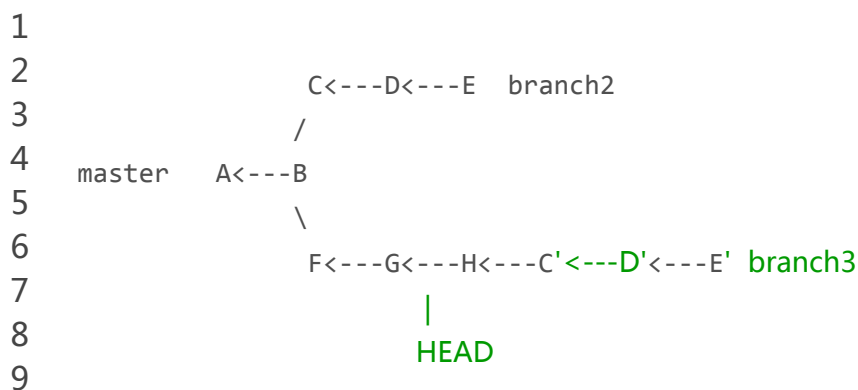
以上两个命令作用相同，表示应用所有提交引入的更改，这些提交是branchname的祖先但不是HEAD的祖先，比如，现在我的仓库中有三个分支，其提交历史如下图：



如果我使用 `git cherry-pick ..branch2` 或者 `git cherry-pick ^HEAD branch2` ,那么会将属于branch2的祖先但不属于branch3的祖先的所有提交引入到当前分支branch3上，并生成新的提交，执行命令如下：

```
1  $ git cherry-pick ..branch2
2  [branch3 c95d8b0] [Description]:branch2  add only-for-branch2
3  Date: Fri Jul 13 20:34:40 2018 +0800
4  1 file changed, 0 insertions(+), 0 deletions(-)
5  create mode 100644 only-for-branch2
6  [branch3 7199a67] [Description]:branch2 modify for only-for-branch2--1
7  Date: Fri Jul 13 20:38:35 2018 +0800
8  1 file changed, 1 insertion(+)
9  [branch3 eb8ab62] [Description]:branch2 modify for only-for-branch2--2
10 Date: Fri Jul 13 20:39:09 2018 +0800
11 1 file changed, 1 insertion(+)
```

执行后的提交历史如下：



常见问题

1.The previous cherry-pick is now empty, possibly due to conflict resolution.

原因:

在 `cherry-pick` 时出现冲突，解决冲突后本地分支中内容和 `cherry-pick` 之前相比没有改变，因此当在以后的步骤中继续 `git cherry-pick` 或执行其他命令时，由于此时还处于上次 `cherry-pick`，都会提示该信息，表示可能是由于解决冲突造成上一次cherry-pick内容是空的。

解决方案:

- 1.执行 `git cherry-pick --abort` 取消上次操作。
- 2.执行 `git commit --allow-empty` ,表示允许空提交。

2.fatal: You are in the middle of a cherry-pick – cannot amend.

原因:

在 `cherry-pick` 时出现冲突，没有解决冲突就执行 `git commit --amend` 命令，从而会提示该信息。

解决方案:

首先在 `git commit --amend` 之前解决冲突，并完成这次 `cherry-pick`：

```
$ git add .  
$ git cherry-pick --continue
```