Git: 특정 commit 으로 이동 후, amend 하기

2014. 2. 28. 16:51

#git rebase #IT

Git: 특정 commit 으로 이동 후, amend 하기

Git 으로 local branch 를 생성 한 후, 특정 source 를 수정함에 있어 여러개의 commit 이 만들어 질 수 있다.
이 때, 기존에 했던 commit 에 추가적인 수정을 하려면 git add/commit --amend 를 하면 되는데 최상위에 있는 commit 은 바로 수정 후, git add/git commit --amend 하면 적용이 되지만 현재 수정하려는 내용이 이전에 있던 commit 에 반영되었으면 하는 경우가 발생한다.

예를 들어, 아래와 같이 commit 들이 있다고 가정하자.

```
commit ec88cc08f5a29f98a3c5f14ca642f235fb1c0fb8
Author: Daeseok Youn <daeseok.youn@gmail.com>
Date:   Fri Feb 28 15:34:22 2014 +0900

    staging: cxt1e1: fix checkpatch errors with open brace '{'

    clean up checkpatch.pl error:
     ERROR: that open brace { should be on the previous line

    Signed-off-by: Daeseok Youn <daeseok.youn@gmail.com>

commit 6007a41fffd430b79775871a2c9eb6c35eb3e6a6
Author: Daeseok Youn <daeseok.youn@gmail.com>
Date:   Fri Feb 28 15:27:51 2014 +0900

    staging: cxt1e1: fix checkpatch error 'assignment in if condition'

    checkpatch.pl error:
     ERROR: do not use assignment in if condition

    Signed-off-by: Daeseok Youn <daeseok.youn@gmail.com>
```

```
commit aa877600a98048d056b65d0c9e7426616e9ccc2f
Author: Daeseok Youn <daeseok.youn@gmail.com>
Date:   Fri Feb 28 15:22:13 2014 +0900

    Staging: cxt1e1: Fix line length over 80 characters in hwprobe.c

    clean up checkpatch.pl warnings:
     WARNING: Line length over 80 characters

    Signed-off-by: Daeseok Youn <daeseok.youn@gmail.com>

commit 2f61d921981a45c687e5dc26d6f7e6a3925ca0e5
Author: Daeseok Youn <daeseok.youn@gmail.com>
Date:   Fri Feb 28 15:14:39 2014 +0900

    staging: cxt1e1: Fix no spaces at the start of a line in hwprobe.c

    clean up checkpatch.pl warnings:
    WARNING: please no spaces at the start of a line in

    Signed-off-by: Daeseok Youn <daeseok.youn@gmail.com>
```

이 때, 추가적인 수정사항이 발생 되었는데, 이 내용이 맨 아래에 있는 "staging: cxt1e1: Fix no spaces at the start of a line in hwprobe.c" 에 적용되었으면 하는 바램이 생겼다.

그렇다면, 일단 그 commit 을 수정할 수 있는 mode(?) 로 가야한다.

```
$ git rebase --interactive 2f61d921981a45c687e5dc26d6f7e6a3925ca0e5^
```

라고 한다. 맨 마지막에 "^" 를 넣어줘야 그 commit 을 포함하여 rebase 를 진행한다.

위에서 처럼 입력하면, 나의 경우는 vim 이 열리면서 아래와 같은 내용이 입력된다.

```
pick 2f61d92 staging: cxt1e1: Fix no spaces at the start of a line in hwprobe.c
pick aa87760 Staging: cxt1e1: Fix line length over 80 characters in hwprobe.c
pick 6007a41 staging: cxt1e1: fix checkpatch error 'assignment in if condition'
pick ec88cc0 staging: cxt1e1: fix checkpatch errors with open brace '{'

# Rebase b9ea35b..ec88cc0 onto b9ea35b
```

```
#
# Commands:
#  p, pick = use commit
#  r, reword = use commit, but edit the commit message
#  e, edit = use commit, but stop for amending
#  s, squash = use commit, but meld into previous commit
#  f, fixup = like "squash", but discard this commit's log message
#  x, exec = run command (the rest of the line) using shell
#
# If you remove a line here THAT COMMIT WILL BE LOST.
# However, if you remove everything, the rebase will be aborted.
#
```

여기서 내가 수정하고 싶은 것은 맨 위에 있는 commit 이다. "pick" 을 "edit" 로 변경하고 저장 하고 나와 git log를 하면
최상위에 그 commit 이 올라와 있다.

이제 원하던 수정을 하자.

수정이 완료되면,

```
$ git add <file>
$ git commit --amend
```

이렇게 하면 일단 현재 원하던 commit 에 추가 반영이 된다.

이제 원래 갖고 있던 commit 들을 복구 해야 한다.

```
$ git rebase --continue
```

이렇게 하면 수정한 commit 이 후에 생성된 commit 들을 merge 하기 시작한다. 하나씩 merge 를 하다가 conflict 날 수 있다.
그러면 rebase 가 중지 되고, conflict 를 해결 한 후에 다시 git rebase --continue 를 입력하라고 한다.

이렇게 해서 마지막으로
Successfully rebased and updated ref/heads/<local branch name>
이라고 나오면 성공한 것이다.