17. REMOVING A COMMIT FROM A BRANCH

Goals

- To learn to delete the branch's latest commits

`Revert` is a powerful command of the previous section that allows you to cancel any commits to the repository. However, both original and cancelled commits are seen in the history of the branch (when using `git log` command).

Often after commit is already made, we realize that it was a mistake. It would be nice to have undo option command allowing deleting incorrect commit immediately. This command would prevent the appearance of unwanted commit in the `git log` history.

The `reset` command

We have already used reset command to match the buffer zone and the selected commit (HEAD commit was used in the previous lesson).

When a commit reference is given (ie, a branch, hash, or tag name), the `reset` command will...

1. Overwrite the current branch so it will point to the correct commit
2. Optionally reset the buffer zone so it will comply with the specified commit
3. Optionally reset the working directory so it will match the specified commit

Check our history

Let us do a quick scan of our commit history.

RUN:

```
git hist
```

RESULT:

```
$ git hist
* 45fa96b 2011-03-09 | Revert "Oops, we didn't want this commit" (HEAD,
master) [Marina Pushkova]
* 846b90c 2011-03-09 | Oops, we didn't want this commit [Marina
Pushkova]
* fa3c141 2011-03-09 | Added HTML header (v1) [Marina Pushkova]
* 8c32287 2011-03-09 | Added standard HTML page tags (v1-beta) [Marina
Pushkova]
* 43628f7 2011-03-09 | Added h1 tag [Marina Pushkova]
* 911e8c9 2011-03-09 | First Commit [Marina Pushkova]
```

We see that the last two commits in this branch are "Oops" and "Revert Oops". Let us remove them with reset command.

Mark this branch first

Let us mark the last commit with tag, so you can find it after removing commits.

RUN:

```
git tag oops
```

Reset commit to previous Oops

At the history log (see above), the commit tagged «v1» is committing previous wrong commit. Let us reset the branch to that point. As the branch has a tag, we can use the tag name in the reset command (if it does not have a tag, we can use the hash value).

RUN:

```
git reset --hard v1
git hist
```

RESULT:

```
$ git reset --hard v1
HEAD is now at fa3c141 Added HTML header
$ git hist
* fa3c141 2011-03-09 | Added HTML header (HEAD, v1, master) [Marina
Pushkova]
* 8c32287 2011-03-09 | Added standard HTML page tags (v1-beta) [Marina
Pushkova]
* 43628f7 2011-03-09 | Added h1 tag [Marina Pushkova]
* 911e8c9 2011-03-09 | First Commit [Marina Pushkova]
```

Our master branch is poiting at commit v1 and "Revert Oops" and "Oops" commits no longer exist in the branch. The `--hard` parameter points out that the working directory must be updated to reflect the new branch head.

Nothing is ever lost

What happens to the wrong commits? They are still in the repository. Actually, we still can refer to them. At the beginning of the lesson, we created «oops» tag for the canceled commit. Let us take a look at *all* commits.

RUN:

```
git hist --all
```

RESULT:

```
$ git hist --all
* 45fa96b 2011-03-09 | Revert "Oops, we didn't want this commit" (oops)
[Marina Pushkova]
* 846b90c 2011-03-09 | Oops, we didn't want this commit [Marina
Pushkova]
* fa3c141 2011-03-09 | Added HTML header (HEAD, v1, master) [Marina
Pushkova]
* 8c32287 2011-03-09 | Added standard HTML page tags (v1-beta) [Marina
Pushkova]
* 43628f7 2011-03-09 | Added h1 tag [Marina Pushkova]
* 911e8c9 2011-03-09 | First Commit [Marina Pushkova]
```

We can see that the wrong commits haven't gone. They are not listed in the master branch anymore but still remain in the repository. They would be still in the repository if we did not tag them, but then we could reference them only by their hash names. Unreferenced commits remain in the repository until the garbage collection software is run by system.

Reset dangers

Resets on local branches are usually harmless. The consequences of any "accident" can be reverted by using the proper commit.

Though, other users sharing the branch can be confused if the branch is shared on remote repositories.

‹ 16. Cancelling commits

18. Removing the oops tag ›