



git

Git tips

Some tips to use `Git`.

Summary

I. Git tips	p. 3
1. Add remote repository on a project and how to use it	p. 3
2. Squash commits	p. 4
3. Rewrite a project history	p. 6
4. Reset a branch on a precise commit	p. 7

I. Git tips

1. Add remote repository on a project and how to use it

It's common to fork a project to contribute on it. However, it can be problematic if you want your master branch (or any other) to be up-to-date since your fork won't update automatically. That's where remote urls get useful.

Add a remote url

Nothing fancy or complicated in here. Once you have your remote url, just add it like this:

```
git remote add the_remote_name your_url
```

Of course, you need to replace "the_remote_name" by the name you want to give to the remote and "your_url" by the url of the remote repository. For example, I want to add a remote url on rust repository:

```
git remote add upstream https://github.com/rust-lang/rust
```

How to use it

Now if I want to update my master branch with the original rust repository:

```
git checkout master # to be sure we're on my local master branch
git fetch upstream # we download the latest changes on the original rust repository
git rebase -i upstream/master # I update my local branch with "upstream" remote url and with its
```

And we're all good. If you want to push changes to a remote repository:

```
git push some_remote_repository HEAD:the_branch_I_want_to_push_on
```

Please note that this is **ALWAYS** better to provide all information to the `git push` command to avoid erroneous pushes.

2. Squash commits

In case you're contributing to a project or just want to do some cleanup in your project commits' history, knowing how to squash is really important.

First step

First, you need to know how many commits you want to squash and where they're located in the commit history. In here, I'll show you the case where you want to squash the latest commits of your history.

For example:

```
> git log
commit 04fa3a3d24e5dc35dc6ab518dac413e9398bc998
Author: Guillaume Gomez <guillaume1.gomez@gmail.com>
Date:    Sun Jul 31 02:17:50 2016 +0200
```

Fixup for pk-config#3

```
commit 04fa3a3d24e78935dc622518dac413e9398bc768
Author: Guillaume Gomez <guillaume1.gomez@gmail.com>
Date:    Sun Jul 31 02:16:59 2016 +0200
```

Fixup for pk-config#2

```
commit 04fa3a3d24e5dc35dc622518dac413e9398bc989
Author: Guillaume Gomez <guillaume1.gomez@gmail.com>
Date:    Sun Jul 31 02:16:27 2016 +0200
```

Fixup for pk-config

```
commit 0cc88a7d344c22e6a61dc2b67be6c63bdc6cb73a
Author: Guillaume Gomez <guillaume1.gomez@gmail.com>
Date:    Sun Jul 31 01:38:34 2016 +0200
```

Add pkg-config to help looking for libs

In here, the 3 last commits (the fixup ones) are too much. We don't want to keep them. So let's squash them into "Add pkg-config to help looking for libs".

git rebase

To remove them from the history, we'll use the `git rebase` command:

```
git rebase -i HEAD~4
```

Let me explain what I did first:

- `-i` is the interactive option.
- `HEAD~4` means (to put it simply) "I want to edit the last four commits starting from the HEAD (which is the last commit of the current branch)".

If you wonder "why 4?", it's because we need to squash all the commits into another one. So
`3` we want to remove + `1` = `4`.

You should now see a screen like the following (if your default git editor is `nano`, otherwise it's just the same):

Git tips

```
Activities Terminal lun. 12:50
imperio@imperio-virtual-machine: ~/rust/video-metadata-rs
imperio@imperio-virtual-machine: ~
imperio@imperio-virtual-machine: ~/rust/git-tips
imperio@imperio-virtual-machine: ~/rust/video-metadata-rs
GNU nano 2.5.3 File: /home/imperio/rust/video-metadata-rs/.git/rebase-merge/git-rebase-todo Modified
pick 360b8bd Add pkg-config to help looking for libs
pick 4d63672 Fixup for pkg-config
pick da1d681 Fixup pkg-config#2
pick 2bec4df Fixup pkg-config#3

# Rebase bc7f450..9a65a09 onto bc7f450 (4 command(s))
#
# 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
# d, drop = remove commit
#
# These lines can be re-ordered; they are executed from top to bottom.
#
# If you remove a line here THAT COMMIT WILL BE LOST.
#
# However, if you remove everything, the rebase will be aborted.
#
# Note that empty commits are commented out
```

Since we need to squash the 3 last ones, replace their "pick" by "squash" (or "s"):

```
Activities Terminal mer. 16:40
imperio@imperio-virtual-machine: ~/rust/video-metadata-rs
imperio@imperio-virtual-machine: ~
imperio@imperio-virtual-machine: ~
imperio@imperio-virtual-machine: ~
imperio@imperio-virtual-machine: ~
imperio@imperio-virtual-machine: ~
GNU nano 2.5.3 File: /home/imperio/rust/video-metadata-rs/.git/rebase-merge/git-rebase-todo Modified
pick 94815bb Add pkg-config to help looking for libs
s 40a784f Fixup for pkg-config
s a6c091f Fixup for pkg-config#2
s 58acc70 Fixup for pkg-config#3

# Rebase bc4cf0c..58acc70 onto bc4cf0c (4 command(s))
#
# 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
# d, drop = remove commit
#
# These lines can be re-ordered; they are executed from top to bottom.
#
# If you remove a line here THAT COMMIT WILL BE LOST.
#
# However, if you remove everything, the rebase will be aborted.
#
# Note that empty commits are commented out
```

Now save and close. `git` will now open a last window which contains the commits' message. Add a `#` at the beginning of every line you don't want:

```
Activities Terminal mer. 16:42
imperio@imperio-virtual-machine: ~/rust/video-metadata-rs
imperio@imperio-virtual-machine: ~
imperio@imperio-virtual-machine: ~
imperio@imperio-virtual-machine: ~
imperio@imperio-virtual-machine: ~
imperio@imperio-virtual-machine: ~
GNU nano 2.5.3 File: /home/imperio/rust/video-metadata-rs/.git/COMMIT_EDITMSG Modified
# This is a combination of 4 commits.
# The first commit's message is:
Add pkg-config to help looking for libs

# This is the 2nd commit message:

#Fixup for pkg-config
# This is the 3rd commit message:

#Fixup for pkg-config#2

# This is the 4th commit message:
#Fixup for pkg-config#3

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Date: Wed Aug 10 16:19:16 2016 +0200
#
# interactive rebase in progress; onto bc4cf0c
# Last commands done (4 commands done):
#   s a6c091f Fixup for pkg-config#2
#   s 58acc70 Fixup for pkg-config#3
# No commands remaining.
# You are currently editing a commit while rebasing branch 'pkg-config' on 'bc4cf0c'.
```

3. Rewrite a project history

Before going any further, I need to warn you: All changes are **definitive**! Once again, be very careful when using this.

Fear git rebase!

Yes, once again, this is `git rebase` which allows to do it. When you're using the interactive option ("-i" for those who didn't read the previous chapter) you can do a lot of things. For example, you can switch the order of a few commits or even delete some! Since there isn't much to tell, I'll provide a quick example:

We start a rebase:

```
git rebase -i HEAD~2
```

We now have the following screen:

```

GNU nano 2.5.3 File: /home/imperio/rust/video-metadata-rs/.git/rebase-merge/git-rebase-todo
pick b7bd300 Add pkg-config to help looking for libs
pick 9a65a09 Some other commit
# Rebase bc4cf0c..9a65a09 onto bc4cf0c (2 command(s))
#
# 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
# d, drop = remove commit
#
# These lines can be re-ordered; they are executed from top to bottom.
#
# If you remove a line here THAT COMMIT WILL BE LOST.
#
# However, if you remove everything, the rebase will be aborted.
#
# Note that empty commits are commented out

```

We switch the two commits:

```

GNU nano 2.5.3 File: /home/imperio/rust/video-metadata-rs/.git/rebase-merge/git-rebase-todo Modified
pick 9a65a09 Some other commit
pick b7bd300 Add pkg-config to help looking for libs
# Rebase bc4cf0c..9a65a09 onto bc4cf0c (2 command(s))
#
# 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
# d, drop = remove commit
#
# These lines can be re-ordered; they are executed from top to bottom.
#
# If you remove a line here THAT COMMIT WILL BE LOST.
#
# However, if you remove everything, the rebase will be aborted.
#
# Note that empty commits are commented out

```

We save and close. Then we need to push the changes on the repository. Once again, I warn you (like I already did in the previous chapter): we'll use the force-push option so be very **careful**!

```
git push -f origin HEAD:the_branch_I_want_to_push_on
```

That's it!

4. Reset a branch on a precise commit

It can happen sometimes that you might want to just remove your last changes. That's where `git reset --hard` comes in.

Get the commit you want to start from

So let's start by finding out what the commit id is:

```
git log
```

Copy the commit id (the highlighted part below):

```

Activities  Terminal
imperio@imperio-virtual-machine: ~/rust/video-metadata-rs

imperio@imperio-virtual-machine: ~$ git log
commit 9a65a09ff4deb4af07a0e9473a2b1292bc415b6b
Author: Guillaume Gomez <guillaume1.gomez@gmail.com>
Date: Wed Aug 10 17:10:19 2016 +0200

    Some other commit

commit b7bd300540f5d0f65bd3c88d9842c52aaff57839
Author: Guillaume Gomez <guillaume1.gomez@gmail.com>
Date: Wed Aug 10 16:19:16 2016 +0200

    Add pkg-config to help looking for libs

commit bc4cf0c4bca43427e271f1c09bd0279a5f175a0c
Merge: 1f51df8 14e9531
Author: Guillaume Gomez <guillaume1.gomez@gmail.com>
Date: Sun Jul 31 02:19:48 2016 +0200

    Merge pull request #19 from GuillaumeGomez/appveyor

commit 14e95314dac4a584a10fedd379c68b7c70ed4f2a
Author: Guillaume Gomez <guillaume1.gomez@gmail.com>
Date: Sun Jul 31 02:12:23 2016 +0200

    Fix url in appveyor

commit 1f51df83b0c634cbb1c55d6e426f0b9804b72df6
Merge: bcf7f450 5fb88a3
Author: Guillaume Gomez <guillaume1.gomez@gmail.com>
Date: Tue Jul 26 18:50:30 2016 +0200

    Add missing double quotes in appveyor script
  
```

Then just reset your local branch on this commit:

```
git reset --hard 1f51df83b0c634cbb1c55d6e426f0b9804b72df6
```

Of course, you need to put your own commit id instead of mine.

If you want/need to push the changes on the repository. Once again, I warn you (like I already did in the previous chapters): we'll use the force-push option so be very **careful**!

```
git push -f origin HEAD:the_branch_I_want_to_push_on
```

That's it!