

2.7 Git Basics - Git Aliases

Git Aliases

Before we finish this chapter on basic Git, there's just one little tip that can make your Git experience simpler, easier, and more familiar: aliases. We won't refer to them or assume you've used them later in the book, but you should probably know how to use them.

Git doesn't automatically infer your command if you type it in partially. If you don't want to type the entire text of each of the Git commands, you can easily set up an alias for each command using `git config`. Here are a couple of examples you may want to set up:

```
$ git config --global alias.co checkout
$ git config --global alias.br branch
$ git config --global alias.ci commit
$ git config --global alias.st status
```

This means that, for example, instead of typing `git commit`, you just need to type `git ci`. As you go on using Git, you'll probably use other commands frequently as well; don't hesitate to create new aliases.

This technique can also be very useful in creating commands that you think should exist. For example, to correct the usability problem you encountered with unstaging a file, you can add your own unstage alias to Git:

```
$ git config --global alias.unstage 'reset HEAD --'
```

This makes the following two commands equivalent:

```
$ git unstage fileA
$ git reset HEAD -- fileA
```

This seems a bit clearer. It's also common to add a `last` command, like this:

```
$ git config --global alias.last 'log -1 HEAD'
```

This way, you can see the last commit easily:

```
$ git last

commit 66938dae3329c7aebc598c2246a8e6af90d04646
Author: Josh Goebel <dreamer3@example.com>
Date: Tue Aug 26 19:48:51 2008 +0800
```

```
test for current head
```

```
Signed-off-by: Scott Chacon <schacon@example.com>
```

As you can tell, Git simply replaces the new command with whatever you alias it for. However, maybe you want to run an external command, rather than a Git subcommand. In that case, you start the command with a `!` character. This is useful if you write your own tools that work with a Git repository. We can demonstrate by aliasing `git visual` to run `gitk`:

```
$ git config --global alias.visual '!gitk'
```

reference : <https://git-scm.com/book/en/v2/Git-Basics-Git-Aliases>