How to list all commits that changed a specific file?

689

162

Is there a way to list all commits that changed a specific file?

git commit

edited Feb 8 at 16:10
Flip
2,414 2 20 47

asked Sep 13 '10 at 14:37
Daniel
3,676 3 13 14

15 Answers

937

The `--follow` works for a particular file

```
git log --follow -- filename
```

Difference to other solutions given

Note that other solutions include `git log path` (without the `--follow`). That approach is handy if you want to track e.g. changes in a **directory**, but stumbles when files were renamed (thus use **`--follow filename`**).

edited Jun 29 '18 at 16:48
Julian
4,952 4 42 80

answered Jan 10 '12 at 18:26
jackrabb1t
9,598 1 16 18

18 +1 `--follow` accounts for renames, so this is more robust than `git log -- path` – Gabe Moothart Aug 7 '13 at 21:09

old `Y` and *new* one. And the opposite, with `--follow` you will get commits regarding that file when it was named `X` *and* when it was named `Y`. – MarSoft Jun 24 '15 at 10:09

5 use "git log --all filename" for view all commits in all branches – Lebnik Aug 13 '15 at 11:22

121

`git log path` should do what you want. From the `git log` man:

```
[--] <path>…

Show only commits that affect any of the specified paths. To prevent confusion with
options and branch names, paths may need to be prefixed with "-- " to separate them
from options or refnames.
```

answered Sep 13 '10 at 14:48
Gabe Moothart
23.7k 12 68 95

10 Does not work if the file's path has changed. jackrabbit's answer does work for this case. – kwahn Apr 3 '14 at 16:09

1 This works if you need to restrict the log to a specific branch – AaronS Jan 17 '17 at 23:02

44

I have been looking at this closely and all these answers don't seem to really show me all the commits across all the branches.

Here is what I have come up with by messing around with the gitk edit view options. This shows me **all the commits for a file** regardless of branch, local, reflog, and remote.

```
gitk --all --first-parent --remotes --reflog --author-date-order -- filename
```

edited Jan 8 '16 at 10:08
Palec

answered Jan 7 '16 at 22:50
BigMiner

3 Also works with `git log` . Very cool. – Stephen Rasku Jun 7 '17 at 14:49

41

Use the command below to get commits for a specific file:

```
git log -p filename
```

edited Nov 16 '16 at 3:40
Community ♦
1 1

answered Apr 2 '13 at 10:41
Sankar Subburaj
3,652 10 39 76

7 I understand that this doesn't exactly answer the question since he wanted a list of commits but this is gold and going in my file. – zkent Jan 7 '16 at 16:04

35

It should be as simple as `git log <somepath>` ; check the manpage (`git-log(1)`).

Personally I like to use `git log --stat <path>` so I can see the impact of each commit on the file.

edited Oct 25 '18 at 0:08
Peter Mortensen
14.2k 19 88 114

answered Sep 13 '10 at 14:50
rfunduk
26.2k 4 55 50

9 Or even `-p` if you want to see the full diff, not just that it had some number of lines modified. – Cascabel Sep 13 '10 at 15:02

True, but that's pretty noisy considering most files have been changed many times over their lives. I don't want to see full diffs of every single commit that ever touched a file. I'm usually looking for a specific thing, so I can get a log with just impacts and then `git show` on the specific commits that look like they matter. – rfunduk Sep 13 '10 at 16:39

git log --stat --follow -- *.html => output list of commits with exactly one files in each commit. Very nice! – Sergio Belevskij Feb 7 at 9:22

```
git log --follow --name-status -- <path>
```

But if you want a more compact list with only what matters:

```
git log --follow --name-status --format='%H' -- <path>
```

or even

```
git log --follow --name-only --format='%H' -- <path>
```

The downside is that `--follow` only works for a single file.

edited Oct 9 '14 at 16:08
user458577

answered Dec 19 '13 at 2:12
Roberto
6,093 9 44 59

4 `--follow` works for a single *path*, which could be a directory. If passed a directory it will run recursively and report changes to all files below that point.
– StvnW Nov 22 '14 at 16:25

11

Alternatively (since Git 1.8.4), it is also possible to just get all the commits which has changed a specific **part** of a file. You can get this by passing the starting line and the ending line number.

The result returned would be the list of commits that modified this particular part. The command goes like:

```
git log --pretty=short -u -L <upperLimit>,<lowerLimit>:<path_to_filename>
```

where `upperLimit` is the `start_line_number` and `lowerLimit` is the `ending_line_number`

More Info - https://www.techpurohit.com/list-some-useful-git-commands

10

If you are trying to **--follow a file deleted** in a previous commit use

```
git log --follow -- filename
```

answered Dec 7 '15 at 15:14
snovelli
3,155 1 20 35

3 For `git` newbies: Use `git log -p --follow -- filename` to display the changes as well. Also note: "filename" can be a file, a directory or a submodule. – Tino May 30 '16 at 13:53

8

If you want to view all the commits that changed a file, in all the branches, use this:

```
git log --follow --all <filepath>
```

edited Oct 25 '18 at 0:15
Peter Mortensen
14.2k 19 88 114

answered Nov 15 '16 at 6:43
Always_Beginner
515 1 11 19

6

If you want to look for all commits by `filename` and **not by** `filepath` , use:

```
git log --all -- '*.wmv'
```

edited Oct 25 '18 at 0:15
Peter Mortensen
14.2k 19 88 114

answered Jan 5 '17 at 4:40
WonderLand
3,392 3 44 62

Palec
7,943 5 43 92

Lebnik
521 7 9

4

If you wish to see all changes made in commits that changed a particular file (rather than just the changes to the file itself), you can pass `--full-diff` :

```
git log -p --full-diff [branch] -- <path>
```

answered Aug 8 '18 at 14:34
Cubic
11.5k 3 33 75

or without the `[branch]` – Anentropic Sep 12 '18 at 17:25

1 @Anentropic The square brackets were supposed to indicate that the argument is optional. – Cubic Sep 12 '18 at 19:39

It's all that I need, It shows full change, includes some change from the merge. – ThanhLD Dec 10 '18 at 4:12

3

```
gitk <path_to_filename>
```

Assuming the package "gitk" is already installed.

If it is not installed, do this:

```
sudo apt-get install gitk
```

And then try the above command. It is for Linux... It might help Linux users if they want a GUI.

edited Oct 25 '18 at 0:11 answered Aug 5 '15 at 13:05

1

On Linux you can use gitk for this.

It can be installed using "sudo apt-get install git-gui gitk". It can be used to see commits of a specific file by "gitk <Filename>".

edited Oct 25 '18 at 0:08
Peter Mortensen
14.2k 19 88 114

answered May 13 '14 at 11:33
Chamila Wijayarathna
994 2 21 37

1

```
# Shows commit history with patch
git log -p -<no_of_commits> --follow <file_name>

# Shows brief details like "1 file changed, 6 insertions(+), 1 deletion(-)"
git log --stat --follow <file_name>
```

Reference

edited Oct 25 '18 at 0:15
Peter Mortensen
14.2k 19 88 114

answered Nov 28 '16 at 12:36
AnshBikram
998 7 8
