In git, is there a way to show untracked stashed files without applying the stash?

Asked 12 years, 2 months ago Modified 1 year, 5 months ago

Viewed 24k times

159

If I run `git stash -u`, I can stash untracked files. However, said untracked files don't show up at all with `git stash show stash@{0}`. Is there any way to show untracked stashed files without applying the stash?

git **git-stash**

Share

Improve this question

Follow

asked Oct 1, 2012 at 21:40

Max Nanasy

6,101 ● 7 ● 35 ● 38

12 Update Q2 2021, 9 years later: `git stash show --include-untracked 0` or even `git stash show --only-untracked 0` would show untracked stashed files without applying the stash. See my answer below. – VonC Mar 27, 2021 at 12:39

7 Answers

Sorted by: Highest score (default)

185

Untracked files are stored in the third parent of a stash commit. (This isn't actually documented, but is pretty obvious from The commit which introduced the -u feature, 787513..., and the way the rest of the documentation for `git-stash` phrases things... or just by doing `git log --graph 'stash@{0}'`)

You can view just the "untracked" portion of the stash via:

```
git show 'stash@{0}^3'
```

or, just the "untracked" tree itself, via:

```
git show 'stash@{0}^3:'
```

or, a particular "untracked" file in the tree, via:

```
git show 'stash@{0}^3:<path/to/file>'
```

There is, unfortunately, no good way to get a summary of the differences between all staged+unstaged+untracked vs "current" state. ie: `git show 'stash@{0}'` cannot be made to include the untracked files. This is because the tree object of the stash commit itself, referred to as `stash@{0}:`, does not include any changes from the third, "unstaged" parent.

This is due to the way stashes are re-applied: tracked files can be easily applied as patches, whereas untracked files can only be applied, in theory, as "whole files".

Share Improve this answer Follow

edited Oct 19, 2021 at 14:04

answered Oct 1, 2012 at 22:12

Will Palmer
5,942 ● 1 ● 29 ● 34

So the parents of the stash commit are (1. Commit stash is made against 2. Index 3. Untracked working copy), and the stash commit itself contains the tracked working copy? `git stash show` appears to show the diff between the working copy and #1 (relevant code from git-stash.sh: `git diff ${FLAGS:---stat} $b_commit $w_commit`, in which $b_commit is #1 and $w_commit is the stash commit); is there any built-in way for `git stash show` to also include #3? – Max Nanasy Oct 1, 2012 at 22:44

6 Note that you get an ugly error (`fatal: ambiguous argument 'stash@{0}^3': unknown revision or path not in the working tree.`) if you *don't* actually have untracked files in that stash (but thought you did). – Randall Jun 8, 2016 at 22:12

2 @antak: nope, `git stash show` does *not* show the untracked files (true for at least git 2.7.4): – Norbert Bérci Mar 31, 2017 at 12:20

2 Note(2.13.2-linux): `git stash pop` will first attempt to restore untracked files, then attempt to restore tracked files. If the latter operation fails(e.g. conflict), the first operation is not rolled-back (untracked-file-stash will stay as is but files are not removed from disk), so even if you fix the conflict, the next pop will fail anyway. – Marinos An Apr 4, 2018 at 8:25

1 In some(?) shell environments `^` is a globbing character. I (am using *zsh* with *zimfw* and) had to escape it like so: `git`

`show stash@{0}\^3` ! – einjohn Sep 11, 2020 at 9:37

28

You can list all stash commits with the following command:

```
git rev-list -g stash
```

Since stashes are represented as a 3-way merge commit of HEAD, the index, and a parent-less "root" commit of untracked files, untracked file stashes can be listed by piping the above output into the following:

```
git rev-list -g stash | git rev-list --stdin --
max-parents=0
```

Useful applications of the above:

Show only untracked, stashed files

```
git rev-list -g stash | git rev-list --stdin --
max-parents=0 | xargs git show --stat
```

Of course, remove the `--stat` *to see the contents of the files.*

Find a specific file

```
git rev-list -g stash | xargs -n1 git ls-tree -r |
sort -u | grep <pattern>
```

Grep untracked files

```
git rev-list -g stash | git rev-list --stdin --
max-parents=0 | xargs git grep <pattern>
```

List all contents of all stashes

```
git rev-list -g stash | git rev-list --stdin |
xargs git show --stat
```

Share Improve this answer

Follow

answered May 17, 2014 at 13:55

Steve

7,078 ● 3 ● 46 ● 42

Thank you! When I use the `Find a specific file` command, I for example get: `100644 blob 05e25619a6f2649b7d635b5aa897cbfc68a4f15d widget/app.css` How can I now pop this changes? (I'm not sure which stash that's in) – John Smith Sep 25, 2020 at 13:29

1 That hash is for the file object (blob), so you could use `git show 05e25619a6f2649b7d635b5aa897cbfc68a4f15d > widget/app.css`. (The `widget/` directory would have to exist, of course, so it isn't as convenient as using `git checkout` with a pathspec.) – Steve Sep 26, 2020 at 16:59

25

> However, said untracked files don't show up at all with `git stash show stash@{0}`.

Note: since Git 2.11 (Q4 2016, 4 years after theis OP), you can reference stash with its index only

```
git stash show 0
```

There was a more recent bug due to `git stash` being rewritten in C, fixed in Git 2.22 (Q2 2019)

> Is there any way to show untracked stashed files without applying the stash?

Why, yes, there is, with Git 2.32 (Q2 2021, 9 years after the OP).

```
git stash show --include-untracked

# or
git config --global stash.showIncludeUntracked true
git stash show
```

"`git stash show`"(man) learned to optionally show untracked part of the stash.

See commit 0af760e, commit d3c7bf7 (03 Mar 2021) by Denton Liu (`Denton-L`).

(Merged by Junio C Hamano -- `gitster` -- in commit f5c73f6, 22 Mar 2021)

> ## `stash show`: teach `--include-untracked` and `--only-untracked`
>
> Signed-off-by: Denton Liu

> Stash entries can be made with untracked files via `git stash push --include-untracked` (man).
>
> However, because the untracked files are stored in the third parent of the stash entry and not the stash entry itself, running `git stash show` (man) does not include the untracked files as part of the diff.
>
> **With `--include-untracked`, untracked paths, which are recorded in the third-parent if it exists, are shown in addition to the paths that have modifications between the stash base and the working tree in the stash.**
>
> It is possible to manually craft a malformed stash entry where duplicate untracked files in the stash entry will mask tracked files.
> We detect and error out in that case via a custom `unpack_trees()` callback: `stash_worktree_untracked_merge()`.

> Also, teach stash the `--only-untracked` option which only shows the untracked files of a stash entry.
> This is similar to `git show stash^3` (man) but it is nice to provide a convenient abstraction for it so that users do not have to think about the underlying implementation.

`git stash` now includes in its man page:

> ```
> 'git stash' show [-u|--include-untracked|-
> -only-untracked] [<diff-options>]
> [<stash>]
> ```

`git stash` now includes in its man page:

> **`--no-include-untracked`**
>
> When used with the `push` and `save` commands, all untracked files are also stashed and then cleaned up with `git clean`.
>
> When used with the `show` command, show the untracked files in the stash entry as part of the diff.
>
> **`--only-untracked`**

> This option is only valid for the `show` command. Show only the untracked files in the stash entry as part of the diff.

And you have a configuration to goes with those new options:

> ## `stash show`: learn
>
> **`stash.showIncludeUntracked`**
>
> Signed-off-by: Denton Liu

> The previous commit teaches `git stash show --include-untracked` (man).
> It may be desirable for a user to be able to always enable the `--include-untracked` behavior.
> Teach the `stash.showIncludeUntracked` config option which allows users to do this in a similar manner to `stash.showPatch`.

`git config` now includes in its man page:

> **`stash.showIncludeUntracked`**
>
> If this is set to true, the `git stash show` command without an option will show the untracked files of a stash entry.

> Defaults to false.

`git stash` now includes in its man page:

> You can use `stash.showIncludeUntracked`, `stash.showStat`, and `stash.showPatch` config variables to change the default behavior.

With Git 2.32 (Q2 2021), the code to handle options recently added to "`git stash show`"(man) around untracked part of the stash **segfaulted** when these options were used on a stash entry that does not record untracked part.

See commit 1ff595d, commit aa2b05d (12 May 2021) by Denton Liu (`Denton-L`).
(Merged by Junio C Hamano -- `gitster` -- in commit a8a2491, 16 May 2021)

> ## `stash show`: fix segfault with `--{include,only}-untracked`
>
> Signed-off-by: Denton Liu

> When `git stash show --include-untracked`(man) or `git stash show --only-untracked`(man) is run on a stash that doesn't include an untracked entry, a segfault occurs.

> This happens because we do not check whether the untracked entry is actually present and just attempt to blindly dereference it.
>
> Ensure that the untracked entry is present before actually attempting to dereference it.

And:

See commit af5cd44 (21 May 2021) by Denton Liu (`Denton-L`).
(Merged by Junio C Hamano -- `gitster` -- in commit 378c7c6, 22 May 2021)

> ## `stash show`: use `stash.showIncludeUntracked` even when `diff` options given
>
> Signed-off-by: Denton Liu

> If options pertaining to how the diff is displayed is provided to `git stash show` (man), the command will ignore the `stash.showIncludeUntracked` configuration variable, defaulting to not showing any untracked files.
> This is unintuitive behaviour since the format of the diff output and whether or not to display untracked files are orthogonal.

> Use `stash.showIncludeUntracked` even when diff options are given.
> Of course, this is still overridable via the command-line options.
>
> Update the documentation to explicitly say which configuration variables will be overridden when a diff options are given.

`git config` now includes in its man page:

> If this is set to `true`, the `git stash show` command will show the untracked files of a stash entry.
>
> Defaults to `false`.

`git stash` now includes in its man page:

> If no `<diff-option>` is provided, the default behavior will be given by the `stash.showStat`, and `stash.showPatch` config variables.
>
> You can also use `stash.showIncludeUntracked` to set whether `--include-untracked` is enabled by default.

Share Improve this answer edited Jul 10, 2023 at 5:38

Follow

answered Mar 27, 2021 at 12:37

VonC

1.3m ● 558 ● 4.7k ● 5.6k

1 NOTE: `--include-untracked` isn't yet in any official `git` release as of May 3 2021, it currently only lives in `master`. Watch for release tags appearing alongside `master` at the bottom of the informational header for this commit: github.com/git/git/commit/… The above is a great answer & deserves all the upvotes once this commit lands. – timoxley May 3, 2021 at 15:12

@timoxley Agreed: this is for Git 2.32 (Q2 2021) – VonC May 3, 2021 at 15:13

1 and a simple way to enable the option with recent git versions is to run: `git config --global stash.showIncludeUntracked true` – Emmanuel Touzery Jul 4, 2023 at 6:29

@EmmanuelTouzery Indeed. I mention that option in this answer as well. – VonC Jul 4, 2023 at 6:35

1 @EmmanuelTouzery I have finally included your comment in the answer! – VonC Jul 10, 2023 at 5:38

16

To list the untracked files in the stash:

```
git ls-tree -r stash@{0}^3 --name-only
```

To show a complete diff of all untracked files (with content):

```
git show stash@{0}^3
```

These commands read the last (most recent) stash. For earlier stashes, increment the number behind the "stash@", for example `stash@{2}` for the second from the last stash.

The reason this works is that `git stash` creates a merge commit for each stash, which can be referenced as `stash@{0}`, `stash@{1}` etc. The first parent of this commit is the HEAD at the time of the stash, the second parent contains the changes to tracked files, and the third (which may not exist) the changes to untracked files.

This is partly explained in the manpage under "Discussion".

Share Improve this answer

Follow

edited Nov 12, 2019 at 17:55

John Kugelman

361k ● 69 ● 546 ● 591

answered Aug 4, 2015 at 1:04

wisbucky

37.6k ● 12 ● 165 ● 112

6

To see all the files in the stash (both tracked and untracked), I added this alias to my config:

```
showstash = "!if test -z $1; then set -- 0; fi;
git show --stat stash@{$1} && git show --stat
```

```
stash@{$1}^3 2>/dev/null || echo No untracked
files -"
```

It takes a single argument of which stash you want to view. Note it will still present it in two back-to-back lists.

The `if...fi` section changes the bash argument $1 to 0 if none was passed.

Share Improve this answer

Follow

edited Nov 12, 2019 at 17:56

John Kugelman

361k ● 69 ● 546 ● 591

answered Jun 8, 2016 at 22:50

Randall

3,024 ● 2 ● 22 ● 28

6

A workaround: Staging files before stashing them will make `git stash show -p` work as expected.

```
git add .
git stash save
```

Note: This way gives the power adding interactive portions too, here is how.
Caution: Ensure you don't have previously staged work, or you won't be able to distinguish it.
This may be of use.

Share Improve this answer

Follow

edited Nov 12, 2019 at 17:56

John Kugelman

361k ● 69 ● 546 ● 591

answered Oct 7, 2016 at 10:11

weshouman
923 ● 1 ● 11 ● 17

0

I had trouble to find and apply the correct stash with my untracked files with git under 2.32.

I used the command in order to find the stash with the untracked files and the date :

```
git rev-list -g stash | git rev-list --stdin --
max-parents=0 | xargs git show --stat
```

After I used another command to show the list of stashes with the date in order to find the correct stash position with the specific date :

```
git stash list --date=local
```

And after, I use again `git stash list` in order to find the name of the stash and apply it with `git stash apply`.

That worked for me. I hope this works for someone else !

Share Improve this answer Follow

edited Jul 7, 2021 at 13:42

answered Jul 7, 2021 at 13:24

Fabien Salles
1,171 16 26