fetch in git doesn't get all branches

Asked 12 years, 5 months ago Modified 3 days ago Viewed 306k times

407

I have cloned a repository, after which somebody else has created a new branch, which I'd like to start working on. I read the manual, and it seems dead straight easy. Strangely it's not working, and all the posts I've found suggest I'm doing the right thing. So I'll subject myself to the lambasting, because there *must* be something obviously wrong with this:

The correct action *seems* to be

```
git fetch
git branch -a
* master
  remotes/origin/HEAD --> origin/master
  remotes/origin/master
git checkout -b dev-gml origin/dev-gml
```

At this point there is a problem, for some reason after `git fetch` I can't see the dev-gml remote branch. Why not? If I clone the repository freshly, it's there, so certainly the remote branch exists:

```
$ mkdir ../gitest
$ cd ../gitest
$ git clone https://github.com/example/proj.git
Cloning into proj...
remote: Counting objects: 1155, done.
remote: Compressing objects: 100% (383/383), done.
remote: Total 1155 (delta 741), reused 1155 (delta
```

```
741)
Receiving objects: 100% (1155/1155), 477.22 KiB |
877 KiB/s, done.
Resolving deltas: 100% (741/741), done.
$ cd projdir
$ git branch -a
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/dev-gml
  remotes/origin/master
```

I've tried `git update`, `git pull`, `git fetch --all`, `git pretty-please` in all possible permutations...

git **git-branch** **git-fetch**

Share

Improve this question

Follow

edited Jan 11, 2021 at 15:12

Jonas

128k ● 100 ● 326 ● 405

asked Jul 24, 2012 at 3:48

Edward Newell

18.4k ● 8 ● 36 ● 37

94 What does `git config --get remote.origin.fetch` produce? If it's not `+refs/heads/*:refs/remotes/origin/*`, it probably should be. – torek Jul 24, 2012 at 5:49

4 yup that's exactly what it produces – Edward Newell Jul 25, 2012 at 5:21

7 Exactly the same problem, but the comment above solved it! I had

`+refs/heads/master:refs/remotes/origin/master` with `master` instead of `*` – Mirko Oct 6, 2012 at 8:43

3 Same problem for me, but none of the suggestions on this page solves it. Weird. – Magnus Jan 28, 2013 at 22:31

3 @thoni56: Yes, this is likely due to a shallow clone. – Trần Việt Hoàng Aug 29, 2017 at 15:14

14 Answers

Sorted by: Highest score (default)

768

The problem can be seen when checking the `remote.origin.fetch` setting
(The lines starting with `$` are bash prompts with the commands I typed. The other lines are the resulting output)

```
$ git config --get remote.origin.fetch
+refs/heads/master:refs/remotes/origin/master
```

As you can see, in my case, the remote was set to fetch the master branch specifically and only. I fixed it as per below, including the second command to check the results.

```
$ git config remote.origin.fetch
"+refs/heads/*:refs/remotes/origin/*"
$ git config --get remote.origin.fetch
+refs/heads/*:refs/remotes/origin/*
```

The wildcard `*` of course means everything under that path.

Unfortunately I saw this comment after I had already dug through and found the answer by trial and error.

Edit: It seems this happens if you do a shallow clone - e.g. `git clone --depth 1 https://github.com/SomeRepo` would cause this problem.

Adding from the comments

> Note that this can happen if you have cloned your repository with a **single branch** only, e.g. git clone --branch --single-branch [] – Narretz

> A shallow clone with git clone --depth **implies --single-branch**, as noted in the man page git-clone(1), so we'd better do it with git clone --depth --no-single-branch . – whatacold

Share Improve this answer

Follow

edited Dec 18 at 12:43

sashoalm

79.2k ● 135 ● 474 ● 816

answered Sep 19, 2014 at 20:25

AndASM

10.2k ● 1 ● 23 ● 33

4 This should probably be the accepted answer, as it actually resolved the issue in the original post. – LocalPCGuy May 26, 2015 at 17:50

8 Note that this can happen if you have cloned your repository with a single branch only, e.g. `git clone <url> --branch <branch> --single-branch [<folder>]` – Narretz May 27, 2016 at 10:34

3 Check stux's answer – Newbee Apr 11, 2018 at 10:17

54 This could happen when you clone with `git clone ... --depth 1` – Anatol Bivol Mar 12, 2019 at 10:26

11 A shallow clone with `git clone --depth <depth> <repo>` implies `--single-branch`, as noted in the man page git-clone(1), so we'd better do it with `git clone --depth <depth> --no-single-branch <repo>`.
– whatacold Nov 30, 2019 at 15:12

273

I had this issue today on a repo.

It wasn't the `+refs/heads/*:refs/remotes/origin/*` issue as per top solution.

Symptom was simply that `git fetch origin` or `git fetch` just didn't appear to do anything, although there were remote branches to fetch.

After trying lots of things, I removed the origin remote, and recreated it. That seems to have fixed it. Don't know why.

remove with: `git remote rm origin`

and recreate with: `git remote add origin <git uri>`

Share Improve this answer answered Mar 9, 2018 at 3:03

Follow

stux
2,936 ● 1 ● 14 ● 6

33 I had correct git config for `remote.origin.fetch` i.e. `+refs/heads/*:refs/remotes/origin/*`. The above solution helped me. – Newbee Apr 11, 2018 at 10:17

21 This solution was the correct one for me also. This is unfortunate since it indicates there's potentially a bug in Git. – Robert Oschler Jun 19, 2018 at 0:56

3 This also solved my issue. I also appear to have this issue on a machine with git version 2.19.1v but didnt experience it on another machine with git version 2.17.1 – jerpint Mar 12, 2019 at 21:23

9 `git remote update origin` worked for me. I guess something needed refreshing? – Felipe Gerard Mar 26, 2019 at 21:15

7 `git remote update origin` didn't work for me but removing and adding the remote did. – Anatoliy Kmetyuk Dec 9, 2019 at 13:25

121

Remote update

You need to run

```
git remote update
```

or

```
git remote update <remote>
```

Then you can run `git branch -r` to list the remote branches.

Checkout a new branch

To track a (new) remote branch as a local branch:

```
git checkout -b <local branch> <remote>/<remote branch>
```

or (sometimes it doesn't work without the extra `remotes/`):

```
git checkout -b <local branch> remotes/<remote>/<remote branch>
```

Helpful git cheatsheets

- Some notes on git
- Git Cheat Sheet (pdf)

Share Improve this answer Follow

edited May 11, 2023 at 19:38

answered Jul 24, 2012 at 4:26

philipvr

6,288 ● 4 ● 33 ● 45

12 But my problem is that I can't checkout an *existing* remote branch, because my git client doesn't think it exists. See my question. Note that when I run `git fetch` followed by `git branch -a` it does not show all the branches. I had to delete my working directory and re-clone to see the branch `dev-gml` that a collaborator made. It worked this time, but we will be branching often! – Edward Newell Jul 25, 2012 at 5:28

Hey @EdwardNewell, thnks for the answer, just to let you know, your link cheat.errtheblog.com/s/git is dead for me... – Kjellski Aug 22, 2013 at 11:51

It's been a long time since I first asked this question, and I just got pinged because someone posted afresh. I'm accepting this answer, even though originally nothing actually worked for me. The reason I have finally marked this correct is because I suspect that what (s)he wrote beside `Edit:` very well might have worked. It is what I would try if I was still facing the problem. HTH – Edward Newell Sep 20, 2014 at 21:33

2 For the record, the bit that helped me here is `git remote update origin`. That made the missing branch visible via `git branch -l -r`. (I did look at `git config --get remote.origin.fetch` and the output was `+refs/heads/*:refs/remotes/origin/*` as expected.) – Robert Dodier Oct 2, 2019 at 19:50

1 This answer also worked for me after I saw my git config for `remote.origin.fetch` was correct. – Britney Smith Jan 12, 2022 at 14:27

Had the same problem today setting up my repo from scratch. I tried everything, nothing worked except

37

removing the origin and re-adding it back again.

```
git remote rm origin
git remote add origin
git@github.com:web3coach/the-blockchain-bar-
newsletter-edition.git

git fetch --all
// Ta daaa all branches fetched
```

Share Improve this answer

Follow

answered May 28, 2020 at 17:45

Lukas Lukac

8,306 ● 10 ● 68 ● 75

10

write it from the terminal

```
git fetch --prune.
```

it works fine.

Share Improve this answer

Follow

answered Jan 3, 2018 at 14:16

Samet ÖZTOPRAK

3,328 ● 3 ● 35 ● 37

2 Thank you! I tried many things and thought I'd just give this a shot... Now to look up what I actually did... – MadTurki Sep 24, 2018 at 15:10

What does it do? – Adam Orłowski Nov 26, 2018 at 11:33

1 It takes all available branches. Look at the head. – Samet ÖZTOPRAK Nov 26, 2018 at 11:39

8

I had cloned the repo with --depth 1, so these answers weren't working. What did work for me was

```
git fetch origin BRANCHNAME:BRANCHNAME
```

It succesfully created a local branch with the same name.

Share Improve this answer

Follow

answered Mar 28, 2023 at 8:49

ninpnin

309 ● 2 ● 4

This doesn't create a tracking branch though. How to do that? Found it stackoverflow.com/questions/23708231/… – Amith Aug 4 at 13:48

6

I had a similar problem, however in my case I could pull/push to the remote branch but `git status` didn't show the local branch state w.r.t the remote ones.

Also, in my case `git config --get remote.origin.fetch` didn't return anything

The problem is that there was a typo in the `.git/config` file in the fetch line of the respective remote block. Probably something I added by mistake previously (sometimes I directly look at this file, or even edit it)

So, check if your remote entry in the `.git/config` file is correct, e.g.:

```
[remote "origin"]
    url = https://[server]/[user or
organization]/[repo].git
    fetch = +refs/heads/*:refs/remotes/origin/*
```

Share Improve this answer Follow

answered Oct 28, 2019 at 9:48

Juh_
15.5k ● 8 ● 61 ● 101

I had quite the same case today and think I found a clue. I cloned my repo, as it contains a ´master´ and a ´devel´ branch. When I run `git config --get remote.origin.fetch` it returns `+refs/heads/master:refs/remotes/origin/master`. But at the same time if i look into the config file it shows **two** refspecs `fetch = +refs/heads/devel:refs/remotes/origin/devel` **as well as** `fetch = +refs/heads/master:refs/remotes/origin/master`. It seems `git config --get remote.origin.fetch` **only returns the last entry**. So, if on doubt look at the file and add the refspec manually. – procra Aug 9, 2021 at 13:20

At least that is what i get in my enviroment. I'm tied on a windows system, so i use the powershell with the posh-git module installed – procra Aug 9, 2021 at 13:24

5

`git checkout --track origin/formats` seemed to do the trick:

```
% git branch      ### show local branches
* main

% git branch - a  ### show local and remote
```

```
branches
* main
  remotes/origin/HEAD -> origin/main
  remote/origin/formats
  remote/origin/main

% git checkout --track origin/formats
Switched to a new branch 'formats'
Branch 'formats' set up to track remote branch
'formats' from 'origin'

% git branch
* formats
  main
```

The following should do the same but with different local branch name:

```
git checkout -b my-formats origin/formats
```

A new syntax `git switch` is available in git c2.23

```
git switch -c <branch> --track <remote>/<branch>
```

Share Improve this answer

Follow

answered Nov 16, 2021 at 4:17

Craig Hicks
2,518 ● 29 ● 40

To make it more specific Create a tracking branch, which means you are now tracking a remote branch.

```
git branch --track branch remote-branch
git branch --track exp remotes/origin/experimental
```

After which you can

```
git branch   # to see the remote tracking branch
"exp" created .
```

Then to work on that branch do

```
git checkout branchname
git checkout exp
```

After you have made changes to the branch. You can git fetch and git merge with your remote tracking branch to merge your changes and push to the remote branch as below.

```
git fetch origin
git merge origin/experimental
git push origin/experimental
```

Hope it helps and gives you an idea, how this works.

edited Jan 26, 2015 at 22:31

Nakilon

35k • 16 • 111 • 146

answered Jul 24, 2012 at 5:26

Swapna

401 ● 2 ● 6

1

This could be due to a face palm moment: if you switch between several clones it is easy to find yourself in the wrong source tree trying to pull a non-existent branch. It is easier when the clones have similar names, or the repos are distinct clones for the same project from each of multiple contributors. A new git clone would obviously seem to solve that "problem" when the real problem is losing focus or working context or both.

Share Improve this answer

Follow

answered Mar 5, 2013 at 15:20

jerseyboy

1,316 ● 13 ● 13

> Thankyou for posting this answer. This was 100% my issue – Nick.Mc Aug 10, 2022 at 10:19

1

My issue: fetch and checkout tell me not found, re-clone works.

`git config --get remote.origin.fetch` output:

```
+refs/heads/*:refs/remotes/origin/*
```

Seems no problem.

Because the commit I try to checkout is on a tag but not a branch, I try to `git fetch --tags`, then checkout the commit, it works finally!

See https://stackoverflow.com/a/14946840/5281824 to configure git to fetch all branches and tags.

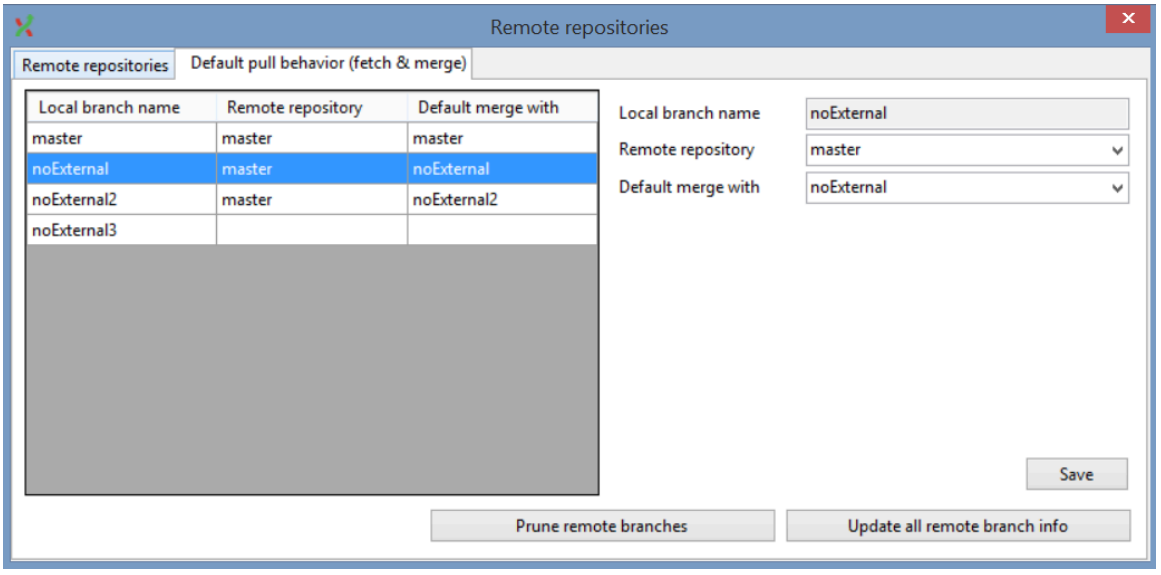
Share Improve this answer Follow

answered Feb 21 at 4:38

weaming
6,395 ● 1 ● 25 ● 16

0

I had to go into my GitExtensions Remote Repositories as nothing here seemed to be working. There I saw that 2 branches had no remote repository configured. after adjusting it looks as follows



Notice branch `noExternal3` still shows as not having a remote repository. Not sure what combo of bash commands would have found or adjusted that.

Share Improve this answer Follow

answered Jul 28, 2014 at 14:15

Maslow

18.7k ● 21 ● 112 ● 194

0

`git remote update --p` **or** `git remote update --prune`

Is a solution.

Share Improve this answer

Follow

answered Dec 16 at 23:36

Malak

401 ● 5 ● 16

-2

All you need to do is, apply the following 2 commands:

```
git fetch --all
```

And once you see the branch (which was not visible before e.g. *osc_at_works*), select that and checkout as below:

```
git checkout origin/team/Enterprise/osc_at_works
```

Share Improve this answer

Follow

answered Apr 28, 2022 at 6:00

Deb

663 ● 5 ● 12

Didn't work for me! – jtlz2 May 20, 2022 at 10:10