How can I specify a branch/tag when adding a Git submodule?

Ask Question

Asked 13 years, 1 month ago Modified 25 days ago Viewed 855k times

1030

How does `git submodule add -b` work?

After adding a submodule with a specific branch, a new cloned repository (after `git submodule update --init`) will be at a specific commit, not the branch itself (`git status` on the submodule shows "Not currently on any branch").

I can't find any information on `.gitmodules` or `.git/config` about the submodule's branch or any specific commit, so how does Git figure it out?

Also, is it possible to specify a tag instead of a branch?

I'm using version 1.6.5.2.

git git-submodules

Share Follow

edited Nov 6, 2018 at 16:20
Fabio says Reinstate Monica
4,868 ●8 ●42 ●60

asked Nov 22, 2009 at 4:55
Ivan
93.4k ●16 ●49 ●58

5 If you have an **existing** submodule which **isn't tracking a branch yet**, but you wish it now would track a branch... see my answer below – VonC Sep 14, 2013 at 7:00

it seems I can specify the branch in the `.gitmodules` file but when I do `git submodule update` and variants (e.g. --all, --recursive, etc) it doesn't pull the git submodule to the right branch. This is obvious rom the `git submodule status`. How do I pull and make sure it's in the right branch? Otherwise what is the point of specifying the branch then? – Charlie Parker Jan 3 at 0:44

Add a comment

14 Answers

Sorted by: Highest score (default)

990

I'd like to add an answer here that is really just a conglomerate of other answers, but I think it may be more complete.

You know you have a Git submodule when you have these two things.

1. Your `.gitmodules` has an entry like so:

```
[submodule "SubmoduleTestRepo"]
    path = SubmoduleTestRepo
    url = https://github.com/jzaccone/SubmoduleTestRepo.git
```

2. You have a submodule object (named SubmoduleTestRepo in this example) in your Git repository. GitHub shows these as "submodule" objects. Or do `git submodule status` from a command line. Git submodule objects are special kinds of Git objects, and **they hold the SHA information for a specific commit.**

Whenever you do a `git submodule update`, it will populate your submodule with content from the commit. It knows where to find the commit because of the information in the `.gitmodules`.

Now, all the `-b` does is add one line in your `.gitmodules` file. So following the same example, it would look like this:

```
[submodule "SubmoduleTestRepo"]
    path = SubmoduleTestRepo
    url = https://github.com/jzaccone/SubmoduleTestRepo.git
    branch = master
```

> **Note:** only branch name is supported in a `.gitmodules` file, but **SHA and TAG are not supported!** (instead of that, the branch's commit of each module can be tracked and updated using "`git add .`", for example like `git add ./SubmoduleTestRepo`, and you do not need to change the `.gitmodules` file each time)

The submodule object is still pointing at a specific commit. The only thing that the `-b` option buys you is the ability to add a `--remote` flag to your update as per Vogella's answer:

```
git submodule update --remote
```

Instead of populating the content of the submodule to the commit pointed to by the submodule, it replaces that commit with the latest commit on the master branch, THEN it populates the submodule with that commit. This can be done in two steps by djacobs7 answer. Since you have now updated the commit the submodule object is pointing to, you have to commit the changed submodule object into your Git repository.

`git submodule add -b` is not some magically way to keep everything up to date with a

The Overflow Blog

Stack Gives Back 2022!

Commit to something big: all about monorepos (Ep. 527)

Featured on Meta

2022 Community-a-thon Recap

Site maintenance - Friday, January 13, 2023 @ 23:00 UTC (6:00 pm EST)

The [shipping] tag is being burninated

Temporary policy: ChatGPT is banned

Linked

1 Bind a specific git submodule branch to a specific main repo branch

0 Why my Git submodule is automatically detached HEAD to a specific commit?

4275 How do I remove a submodule?

1034 Definition of "downstream" and "upstream"

465 Why did my Git repo enter a detached HEAD state?

233 Why is my Git Submodule HEAD detached from master?

193 How do I check out a specific version of a submodule using 'git submodule'?

176 git submodule tracking latest

193 SVN:externals equivalent in Git?

88 Force Git submodules to always stay current

See more linked questions

Related

7867 How do I remove local (untracked) files from the current Git working tree?

10833 How do I undo 'git add' before commit?

25257 How do I undo the most recent local commits in Git?

9128 How do I force "git pull" to overwrite local files?

4275 How do I remove a submodule?

6210 Move the most recent commit(s) to a new branch with Git

8230 How do I check out a remote Git branch?

19843 How do I delete a Git branch locally and remotely?

2174 How can I git stash a specific file?

11035 How do I rename a local Git branch?

Hot Network Questions

Can I use a tantalum capacitor instead of an aluminium capacitor with an AP1509-50SG?

What's going on with D&D's leaked OGL 1.1? How does it affect me?

Conjecture: If circular coins of any sizes are in a convex polygonal frame, with each coin touching exactly one edge, then all the coins can move

Did the Finns have aluminum screw caps in 1939?

Title Explanation of the movie "The Pale Blue Eye"

Select data from a structured list

populating it.

Share Follow

edited Sep 22, 2022 at 20:42

answered Sep 14, 2013 at 2:33
Johnny Z
13.5k ●4 ●27 ●35

34 This answer should have more up-votes. I've been reading many posts for the past day and this clears up all the confusion. Coming from the SVN world and using externals - one wants to believe that git submodule branch tracking does magically keep everything up to date from the branch - but this is not true! You have to explicitly update them! As you mention, you must commit changed submodule objects. – dtmland Mar 25, 2015 at 16:16

20 Does this branch tracking also work with **tags**? Instead of a branch I specified a tag in my `.gitmodules` and after doing `$ git submodule update --init --remote TestModule` I got an error saying `fatal: Needed a single revision` and `Unable to find current origin/TestTag revision in submodule path 'TestModule'`. When doing it with a real branch it works. Is there anyway to specify a tag in `.gitmodules` without having to specify the exact commit? – Hhut Sep 4, 2015 at 8:19

9 This doesn't seem to work. I updated the hash in `.gitmodules` and ran `git submodule update` and nothing happened? – CMCDragonkai Apr 4, 2016 at 14:17

3 Somehow this doesnt work for me. With a SHA Commit Id, I always get an error "Unable to find current revision (I double checked the revision number of HEAD and its correct) . However if I use master it works. – infoclogged Jun 29, 2016 at 15:27

3 Entering a SHA into the branch attribute doesn't work for me either. This usage is also *not* supported by the docs: git-scm.com/docs/gitmodules – Jakub Bochenski Jun 28, 2017 at 14:44

Show **13** more comments

918

Note: Git 1.8.2 added the possibility to track branches. See some of the answers below.

It's a little confusing to get used to this, but submodules are not on a branch. They are, like you say, just a pointer to a particular commit of the submodule's repository.

This means, when someone else checks out your repository, or pulls your code, and does git submodule update, the submodule is checked out to that particular commit.

This is great for a submodule that does not change often, because then everyone on the project can have the submodule at the same commit.

If you want to move the submodule to a particular tag:

```
cd submodule_directory
git checkout v1.0
cd ..
git add submodule_directory
git commit -m "moved submodule to v1.0"
git push
```

Then, another developer who wants to have submodule_directory changed to that tag, does this

```
git pull
git submodule update --init
```

`git pull` changes which commit their submodule directory points to. `git submodule update` actually merges in the new code.

Share Follow

edited Jul 2, 2019 at 21:43
Julius Smith
3 ●2

answered Nov 22, 2009 at 9:19
djacobs7
11k ●3 ●24 ●32

13 That's a very good explanation, thanks! And of course, after reading your answer, I realized the commit is saved inside the submodule itself (submodule/.git/HEAD). – Ivan Nov 22, 2009 at 18:19

6 This doesn't seem to work on git 1.7.4.4. `cd my_submodule; git checkout [ref in submodule's repository` yields `fatal: reference is not a tree: ...`. It's as if `git` will only operate on the parent repository. – James A. Rosen May 4, 2012 at 21:12

4 It's good to use git submodules even for projects that are updated often. The linux kernel uses it and it isn't so bad – user9903 Apr 30, 2013 at 18:15

15 Is `git checkout v1.0` a branch or a tag? – Bernhard Döbler Oct 1, 2015 at 22:30

13 Consider a tag a human readable alias to a commit. And a commit is a set of specific state for each file. A branch is essentially the same thing except you can make changes to it. – deadbabykitten Feb 2, 2016 at 0:13

Show **6** more comments

457

(Git 2.22, Q2 2019, has introduced `git submodule set-branch --branch aBranch -- <submodule_path>`)

Note that **if you have an *existing* submodule which *isn't* tracking a branch yet**, then (if you have git 1.8.2+):

- Make sure the parent repo knows that its submodule now tracks a branch:

How to do graphic design work for free?
Gollum wanted possession of the Ring but did not want to wear it?
Is there any primary source for Mussolini being called "Juliet"?
Adjective for questions that demonstrate good understanding
Is a 401K taxed as regular income when the owner dies?
How will this static initiative houserule affect game balance?
Is it correct to write "this book discusses"?
How to find ESR in capacitor datasheet?
Time domain to Frequency domain: Simple question
How might a flippered being wear a watch?
I want to write a LaTeX document in Yiddish
Is there a work-around for a lack of clearance to use my torque-wrench?
Why "looking-back" does not match "[\n]+"?
Convert Range Hood Light Switch to High/Off/Low
Is it possible to use fake backends to run Qiskit Runtime primitives?
Why does bleeding radiators save energy?
How to improve Scrum Master performance
Question feed

- Make sure your submodule is actually at the latest of that branch:

```
cd path/to/your/submodule
git checkout -b branch --track origin/branch
  # if the master branch already exist:
  git branch -u origin/master master
```

(with 'origin' being the name of the **upstream remote repo** the submodule has been cloned from.
A `git remote -v` inside that submodule will display it. Usually, it is 'origin')

- Don't forget to record the new state of your submodule in your parent repo:

```
cd /path/to/your/parent/repo
git add path/to/your/submodule
git commit -m "Make submodule tracking a branch"
```

- Subsequent update for that submodule will have to use the `--remote` option:

```
# update your submodule
# --remote will also fetch and ensure that
# the latest commit from the branch is used
git submodule update --remote

# to avoid fetching use
git submodule update --remote --no-fetch
```

Note that with Git 2.10+ (Q3 2016), you can use '`.`' as a branch name:

> The name of the branch is recorded as `submodule.<name>.branch` in `.gitmodules` for `update --remote`.
> **A special value of `.` is used to indicate that the name of the branch in the submodule should be the same name as the current branch in the current repository**.

But, as commented by LubosD

> With `git checkout`, if the branch name to follow is "`.`", it will kill your uncommitted work!
> Use `git switch` instead.

That means Git 2.23 (August 2019) or more.

See "Confused by `git checkout`"

If you want to update all your submodules following a branch:

```
    git submodule update --recursive --remote
```

Note that the result, for each updated submodule, will **almost always be a detached HEAD**, as Dan Cameron note in his answer.

(Clintm notes in the comments that, if you run `git submodule update --remote` and the resulting sha1 is the same as the branch the submodule is currently on, it won't do anything and leave the submodule still "on that branch" and not in detached head state.)

To ensure the branch is actually checked out (and that won't modify the SHA1 of the **special entry** representing the submodule for the parent repo), he suggests:

```
git submodule foreach -q --recursive 'branch="$(git config -f $toplevel/.gitmodules s
```

Each submodule will still reference the same SHA1, but if you do make new commits, you will be able to push them because they will be referenced by the branch you want the submodule to track.
After that push within a submodule, don't forget to go back to the parent repo, add, commit and push the new SHA1 for those modified submodules.

Note the use of `$toplevel`, recommended in the comments by Alexander Pogrebnyak.
`$toplevel` was introduced in git1.7.2 in May 2010: commit f030c96.

> it contains the absolute path of the top level directory (where `.gitmodules` is).

`dtmland` adds in the comments:

> The foreach script will fail to checkout submodules that are not following a branch.
> However, this command gives you both:

```
git submodule foreach -q --recursive 'branch="$(git config -f $toplevel/.gitmodules
git submodule foreach -q --recursive \
    'branch="$(git config -f $toplevel/.gitmodules submodule.$name.branch)"; \

     [ "$branch" = "" ] && \
     git checkout master || git switch $branch' –
```

---

umläute refines dtmland's command with a simplified version in the comments:

```
git submodule foreach -q --recursive 'git switch $(git config -f $toplevel/.gitmodule
```

multiple lines:

```
git submodule foreach -q --recursive \
  'git switch \
  $(git config -f $toplevel/.gitmodules submodule.$name.branch || echo master)'
```

Before Git 2.26 (Q1 2020), a fetch that is told to recursively fetch updates in submodules inevitably produces reams of output, and it becomes hard to spot error messages.

The command has been taught to enumerate submodules that had errors at the end of the operation.

See commit 0222540 (16 Jan 2020) by Emily Shaffer (`nasamuffin`).
(Merged by Junio C Hamano -- `gitster` -- in commit b5c71cc, 05 Feb 2020)

> ## `fetch`: emphasize failure during submodule fetch
>
> Signed-off-by: Emily Shaffer

> In cases when a submodule fetch fails when there are many submodules, the error from the lone failing submodule fetch is buried under activity on the other submodules if more than one fetch fell back on `fetch-by-oid`.
> **Call out a failure late so the user is aware that something went wrong, and where**.

> Because `fetch_finish()` is only called synchronously by `run_processes_parallel,` mutexing is not required around `submodules_with_errors`.

Note that, with Git 2.28 (Q3 2020), Rewrite of parts of the scripted "git submodule" Porcelain command continues; this time it is "`git submodule set-branch`" subcommand's turn.

See commit 2964d6e (02 Jun 2020) by Shourya Shukla (`periperidip`).
(Merged by Junio C Hamano -- `gitster` -- in commit 1046282, 25 Jun 2020)

> ## `submodule`: port subcommand 'set-branch' from shell to C
>
> Mentored-by: Christian Couder
>
> Mentored-by: Kaartic Sivaraam
>
> Helped-by: Denton Liu
>
> Helped-by: Eric Sunshine
>
> Helped-by: Đoàn Trần Công Danh
>
> Signed-off-by: Shourya Shukla

> Convert submodule subcommand 'set-branch' to a builtin and call it via `git submodule.sh`.

Share Follow

edited Jun 28, 2020 at 2:16

answered Sep 14, 2013 at 6:59

VonC
1.2m ● 505 ● 4232 ● 5040

2 Question: if i have folder subModule1 and wish to track master branch, would the resulting command look like this: **git config -f .gitmodules submodule.subModule1.branch master** – BraveNewMath Sep 17, 2013 at 20:41

1 The `foreach` script will not depend on the hardcoded `<path>`, if you substitute `<path>` with `$toplevel/`. – Alexander Pogrebnyak Nov 27, 2013 at 18:19

1 The `foreach` script will fail to checkout submodules that are not following a branch. However, this command gives you both: `git submodule foreach -q --recursive 'branch="$(git config -f $toplevel/.gitmodules submodule.$name.branch)"; [ "$branch" = "" ] && git checkout master || git checkout $branch'` – dtmland Jul 10, 2015 at 19:09

2 here's a simplified version of @dtmland's script: `git submodule foreach -q --recursive 'git checkout $(git config -f $toplevel/.gitmodules submodule.$name.branch || echo master)'` – umläute Oct 15, 2015 at 11:54

1 Ohh! Actually the foreach script is unnecessary. We have to execute the submodule update with the --merge

Show **19** more comments

303

Git 1.8.2 added the possibility to track branches.

```
# add submodule to track branch_name branch
git submodule add -b branch_name URL_to_Git_repo optional_directory_rename

# update your submodule
git submodule update --remote
```

See also Git submodules

Share Follow

edited Dec 5, 2020 at 1:19
DDS
2,187 ●15 ●30

answered Apr 3, 2013 at 8:44
vogella
23.4k ●4 ●29 ●26

11 Does this apply to tags as well? – ThorSummoner Jul 2, 2014 at 21:28

1 How does adding submodule in such way reflect on `.gitmodules` file? – Eugene Jul 20, 2014 at 12:22

2 Thanks I just used the info about to help me to create a submodule folder that is synced with a GitHub gh-pages website: full example at github.com/o2platform/fluentnode/issues/22 – Dinis Cruz Dec 24, 2014 at 2:03

14 You can lock to a *tag* with `git submodule add -b tags/<sometag> <url>` which you can see as the line `branch = tags/<sometag>` in `.gitmodules` – KCD Oct 16, 2015 at 2:27

15 @KCD Which version of git can do that with tags. Mine doesn't work? – CMCDragonkai Apr 4, 2016 at 13:56

Show **3** more comments

75

An example of how I use Git submodules.

1. Create a new repository
2. Then clone another repository as a submodule
3. Then we have that submodule use a tag called V3.1.2
4. And then we commit.

And that looks a little bit like this:

```
git init
vi README
git add README
git commit
git submodule add git://github.com/XXXXX/xxx.yyyy.git stm32_std_lib
git status

git submodule init
git submodule update

cd stm32_std_lib/
git reset --hard V3.1.2
cd ..
git commit -a

git submodule status
```

Maybe it helps (even though I use a tag and not a branch)?

Share Follow

edited Dec 30, 2018 at 10:43
Peter Mortensen
30.8k ●21 ●104 ●125

answered Nov 22, 2009 at 9:53
Johan
19.7k ●28 ●92 ●110

5 It's basically the same answer as djacobs7, but thanks anyway :) – Ivan Nov 22, 2009 at 18:22

1 Should you be able to commit a change after your `git reset --hard V3.1.2`? I just get a "nothing to commit" with a `git status` of the parent directory. – Nick Radford Oct 10, 2012 at 20:07

1 @Ivan: Could you explain how this is the same as djacobs7's response? As far as I see, his response doesn't even include the 'submodule add' command, instead the repo is added directly, without any link to the module's original git repo. At least when I tried this approach there was no link in .gitmodules. – Michel Müller Aug 12, 2013 at 8:14

djacobs7's response doesn't include the whole explanation starting from adding the submodule. He assumes you already have it. – CodeMonkey Aug 22, 2017 at 6:04

doesn't it just add the entire submodule contents as tracked objects to your main repo? – Sergey Grechin Apr 30, 2020 at 7:58

Add a comment

43

In my experience switching branches in the superproject or future checkouts will still cause detached HEADs of submodules regardless if the submodule is properly added and tracked (i.e. @djacobs7 and @Johnny Z answers).

And instead of manually checking out the correct branch manually or through a script git submodule foreach can be used.

```
git submodule foreach -q --recursive 'branch="$(git config -f $toplevel.gitmodules
submodule.$name.branch)"; git checkout $branch'
```

Share Follow

edited May 19, 2021 at 15:49
Diogo Cardoso
21k ●26 ●98 ●138

answered Nov 9, 2013 at 4:54
Dan Cameron
756 ●6 ●8

Nice. +1. I have included your command in my answer. – VonC Nov 9, 2013 at 8:17

Add a comment

41

Git submodules are a little bit strange - they're always in "detached head" mode - they don't update to the latest commit on a branch like you might expect.

This does make some sense when you think about it, though. Let's say I create repository *foo* with submodule *bar*. I push my changes and tell you to check out commit a7402be from repository *foo*.

Then imagine that someone commits a change to repository *bar* before you can make your clone.

When you check out commit a7402be from repository *foo*, you expect to get the same code I pushed. That's why submodules don't update until you tell them to explicitly and then make a new commit.

Personally I think submodules are the most confusing part of Git. There are lots of places that can explain submodules better than I can. I recommend Pro Git by Scott Chacon.

Share Follow

edited Oct 19, 2018 at 20:06
Peter Mortensen
30.8k ●21 ●104 ●125

answered Nov 22, 2009 at 5:22
Neall
25.7k ●5 ●49 ●48

I think it's time I start reading some git books, thanks for the recommendation. – Ivan Nov 22, 2009 at 18:21

Sorry, but you didn't clarify if one would get the same as you pushed to a7402be , or get the latest of bar, though your version of foo. Thanks :) – mjs Dec 1, 2011 at 12:44

7 The issue is that there should be an option to say "keep this submodule on branch X" so that if you WANT it to automatically update itself then you can make that happen. It would make submodules much more useful for managing e.g. a WordPress installation where plugins are all Git repos without having to re-save the superproject for every plugin that updates. – jerclarke Oct 25, 2012 at 18:23

@jeremyclark `git clone git://github.com/git/git.git` and push that feature...? =D – Alastair Nov 7, 2012 at 4:01

@0xC0000022L That is because getting the finger from `git` 10000000 times a day is still preferrable to the beastiality other versioning tools deem sensible. – user1129682 Sep 2, 2019 at 10:55

Show **1** more comment

40

Existing answers have the second step missing and overloaded with details.

To switch an existing submodule to track a new remote url and/or a new branch:

1. Edit the source of truth in `.gitmodules`.

For example, from

```
[submodule "api"]
    path = api
    url = https://github.com/<original_repo>/api.git
```

to

```
[submodule "api"]
    path = api
    url = https://github.com/<another_repo>/api.git
    branch = work-in-progress
```

You can also use `hexsha` for a commit hash. Or `tag`, but see 3.

2. `git submodule sync`: Updates the description of submodules cached by git in `.git/modules` from the just-edited source of truth specified in `.gitmodules`.
3. *Only if you specify a tag: `git submodule foreach --recursive 'git fetch --tags'` to fetch tags.*
4. `git submodule update --init --recursive --remote`: Updates the checked-out submodules in the working copy.
5. Commit the changes.

Share Follow

edited Sep 26, 2022 at 20:21

answered Feb 8, 2022 at 0:15
Dmitry Spikhalskiy
5,067 ●1 ●25 ●38

2 Thanks for clear solution. – hobbydev Mar 7, 2022 at 4:59

Add a comment

25

- `cd` to root of your repository containing the submodules
- Open `.gitmodules` for editing
- Add line below `path = ...` and `url = ...` that says `branch = your-branch`, for each submodule; save file `.gitmodules`.
- then without changing directory do `$ git submodule update --remote`

...this should pull in the latest commits on the specified branch, for each submodule thus modified.

Share Follow

edited Oct 19, 2018 at 20:14
Peter Mortensen
30.8k ● 21 ● 104 ● 125

answered Jun 12, 2016 at 13:34
Engineer
8,303 ● 7 ● 63 ● 101

Add a comment

16

I have this in my `.gitconfig` file. It is still a draft, but proved useful as of now. It helps me to always reattach the submodules to their branch.

```
[alias]

######################
#
# Submodules aliases
#
######################

# git sm-trackbranch: places all submodules on their respective branch specified in .
# This works if submodules are configured to track a branch, i.e if .gitmodules looks
# [submodule "my-submodule"]
#   path = my-submodule
#   url = git@wherever.you.like/my-submodule.git
#   branch = my-branch
sm-trackbranch = "! git submodule foreach -q --recursive 'branch=\"$(git config -f $t

# sm-pullrebase:
# - pull --rebase on the master repo
# - sm-trackbranch on every submodule
# - pull --rebase on each submodule
#
# Important note:
# - have a clean master repo and subrepos before doing this!
# - this is *not* equivalent to getting the last committed
#   master repo + its submodules: if some submodules are tracking branches
#   that have evolved since the last commit in the master repo,
#   they will be using those more recent commits!
#
#   (Note: On the contrary, git submodule update will stick
#   to the last committed SHA1 in the master repo)
sm-pullrebase = "! git pull --rebase; git submodule update; git sm-trackbranch ; git

# git sm-diff will diff the master repo *and* its submodules
sm-diff = "! git diff && git submodule foreach 'git diff' "
```

Share Follow

edited Sep 24, 2021 at 8:36
Lucas
490 ● 2 ● 12 ● 16

answered Apr 1, 2015 at 16:27
Pascal T.
3,776 ● 4 ● 33 ● 35

Add a comment

4

We use Quack to pull a specific module from another Git repository. We need to pull code without the whole code base of the provided repository - we need a very specific module / file from that huge repository and should be updated every time we run update.

So we achieved it in this way:

Create configuration

```
name: Project Name

modules:
  local/path:
    repository: https://github.com/<username>/<repo>.git
    path: repo/path
    branch: dev
  other/local/path/filename.txt:
    repository: https://github.com/<username>/<repo>.git
    hexsha: 9e3e9642cfea36f4ae216d27df100134920143b9
    path: repo/path/filename.txt

profiles:
  init:
    tasks: ['modules']
```

With the above configuration, it creates one directory from the provided GitHub repository as specified in first module configuration, and the other one is to pull and create a file from the given repository.

Other developers just need to run

```
$ quack
```

Share Follow

edited Oct 19, 2018 at 20:14
Peter Mortensen
30.8k ●21 ●104 ●125

answered Oct 22, 2015 at 14:35
Love Sharma
1,969 ●1 ●18 ●37

Add a comment

4

The only effect of choosing a branch for a submodule is that, whenever you pass the `--remote` option in the `git submodule update` command line, Git will check out in *detached HEAD* mode (if the default `--checkout` behavior is selected) the latest commit of that selected **remote** branch.

You must be particularly careful when using this remote branch tracking feature for Git submodules if you work with shallow clones of submodules. The branch you choose for this purpose in submodule settings **IS NOT** the one that will be cloned during `git submodule update --remote`. If you pass also the `--depth` parameter and *you do not instruct Git about which branch you want to clone* -- **and actually you cannot** in the `git submodule update` command line!! -- , it will implicitly behave like explained in the `git-clone(1)` documentation for `git clone --single-branch` when the explicit `--branch` parameter is missing, and therefore **it will clone the primary branch only**.

With no surprise, after the clone stage performed by the `git submodule update` command, it will finally try to check out the latest commit for the **remote** branch you previously set up for the submodule, and, if this is not the primary one, it is not part of your local shallow clone, and therefore it will fail with

> fatal: Needed a single revision
>
> Unable to find current origin/***NotThePrimaryBranch*** revision in submodule path 'mySubmodule'

Share Follow

edited Jun 20, 2020 at 9:12
Community Bot
1 ●1

answered May 29, 2018 at 15:52
LuKePicci
51 ●3

Add a comment

4

git submodule add -b develop --name branch-name -- https://branch.git

Share Follow

answered Mar 7, 2019 at 9:04
Passiondroid
1,555 ●1 ●16 ●28

No need to downvote. Actually, there's some reasoning behind this. This actually worked for me on Windows with GitBash for something like this: `git submodule add -b master --name master -- https://github.com/nlohmann/json.git libs/json` Cheers for the tip! – anatoli Jun 11, 2020 at 13:28

I did it in the config file, did not find the corresponding command, thanks ! – Mat M May 29, 2022 at 20:23

Add a comment

-1

Use the below commands

To add submodule (branch-branch)

```
git submodule add -b stage git@github.optum.com:orgname/${reponame}.git
```

Update Submodule (branch-branch)

```
######Clone########

> git clone -b master --single-branch --recurse-submodules git@github.com:orgname/pro
or
> git clone -b stage --single-branch --recurse-submodules git@github.com:orgname/proj

######Update#######
> git submodule update  --remote (only for remote branch -ie master)
or
> git submodule update --recursive --remote

> git submodule update --init --recursive (for remaining branchs)

if you get fatal need single revision error then use below commands:-

before:-

*** stage
  remotes/origin/stage**

git branch -a

git symbolic-ref refs/remotes/origin/HEAD refs/remotes/origin/{branch name}
ex:- git symbolic-ref refs/remotes/origin/HEAD refs/remotes/origin/stage

after:-
```

```
then

> git reset --hard
```

after commit and push accordingly.. now submodules will get updated.

Share Follow

answered Dec 19, 2022 at 13:30
Marshal Sriraj
39 ● 4

Add a comment

Your Answer

Sign up or log in

Sign up using Google

Sign up using Facebook

Sign up using Email and Password

Post as a guest

Name

Email
Required, but never shown

Post Your Answer *By clicking “Post Your Answer”, you agree to our terms of service, privacy policy and cookie policy*

Not the answer you're looking for? Browse other questions tagged git git-submodules or ask your own question.

STACK OVERFLOW
Questions
Help

PRODUCTS
Teams
Advertising
Collectives
Talent

COMPANY
About
Press
Work Here
Legal
Privacy Policy
Terms of Service
Contact Us
Cookie Settings
Cookie Policy

STACK EXCHANGE NETWORK
Technology
Culture & recreation
Life & arts
Science
Professional
Business

API
Data

Blog Facebook Twitter LinkedIn Instagram