How to use git?

For people not familiar with git.

What is the advantage of Git over another version control system, e.g SVN?

In short terms, git is a distributed version control system (compared to svn which is a centralised system). In this way, each developer has a their own local repository, complete with a full history of commits, instead of a working copy. This also has the advantage that the production branch cannot be broken, each developer work on their own local repository. This creates a more reliable environment.

A detailed explanation can be seen here.

These are the most simple commands. For more details, please consult the git documentation.

Try this: https://try.github.io/. It teaches you already 95% of the git commands you will ever need.

To be able to do pull requests on stash, access with your DESY account need to be added. Please request it at it-atlassian@desy.de.

Git configuration

In your *~/.gitconfig*

```
[alias]
        st = status
        ci = commit
        co = checkout
        lola = log --graph --decorate=full\n--pretty=oneline --abbrev-commit --all
        logf = "!echo \"Remember to add -S<string>\" ; git log --color -p --source --all"
  [color]
        ui = true
        diff = auto
        status = auto
  [core]
        excludesfile = <YourHomeFolder>/.gitignore_global
        editor = emacs -nw
  [branch]
        autosetuprebase = always
  [push]
          ## for newer versions of git, otherwise try "simple"
        default = matching
```

This will defines some useful alias. Turns on color for some commands. It defines a global ignore file that you will need to create yourself. Sets the default editor to emacs in `No Window` mode. The `autosetuprebase = always` makes every pull a rebase operation. "default matching" means pushes are done to matching branches.

Cloning a repository

Clone

```
git clone https://stash.desy.de/scm/calice/<repo>
```

This will create a local working copy of the repository. You can also use ssh to clone the repository via *https://<username>@stash.desy.de/scm/calice/<repo>.* More details can be seen here.

Get a list of branches or tags:

List branches and tags

```
calice_userlib$ git branch -vv
* master 3e07eae [origin/master] Merge pull request #1 in CALICE/calice_userlib from feature to master

git tag
```

This gives a full list of the branches with the last commit comment and the list of the tags.

Workflow

Make a fork

Create a fork of the software by clicking on "Create Fork". This will create a copy of the repository on your personal account and it will stay in synced.

Add a downstream remote pointing to your fork

Downstream

```
git remote add downstream https://<username>@stash.desy.de/scm/~<username>/calice_userlib.git
```

You should get sth like that:

```
$ git remote -vv
downstream https://ebrianne@stash.desy.de/scm/~ebrianne/calice_userlib.git (fetch)
downstream https://ebrianne@stash.desy.de/scm/~ebrianne/calice_userlib.git (push)
origin https://ebrianne@stash.desy.de/scm/calice/calice_userlib.git (fetch)
origin https://ebrianne@stash.desy.de/scm/calice/calice_userlib.git (push)
```

Working, updating

To develop new features, it is always advised to create a new branch to work on. If there are commit added to the `master`, one can incorporate them later via a `rebase` command.

First make sure you have the latest `origin`

Fetch

```
git fetch origin
```

The `git fetch` command is the 'safe' version for downloading new content without updating the local working branch. If you are behind the `master` branch, you ll need to do

Rebase branch

```
git checkout master
git pull
```

If you are using the configuration as described above (`autosetuprebase=always`), this will `rebase` your local `master` branch to the remote `maste` branch. If you have local changes that are not committed pull (i.e., `rebase`) will not be allowed.

Use `git stash` to store the local changes without committing them.

Make sure your are not ahead of the master branch!

Now you can create your local new branch

Create a branch

```
git checkout -b <branchname>
```

This will create a new branch call `<branchname>` based from the current branch `master.` Now you can work on it and commit!

To get to any branch, just do:

Checkout a branch

```
git checkout <branchname>
```

Staging and commit

To stage track and untracked files:

Add file

```
git add <yourfile> or <directory>
```

Get status of the tracked files:

Status

```
git status
```

This gives the the list of tracked and untracked files in the working branch.

Commit to the local branch:

Commit

```
git commit -m "<Comments>"
```

This will commit the staged files to the working branch.

Merging and deleting branches

Once you are happy with the features you have developed in your developing branch, you might decide to merge them with your master branch and delete the branch you used for feature developing.

Before you merge your branch with the master, you might need to check whether your master is up-to-date (see below 'Before a pull request').

To merge branch with master, first go to master branch, then merge:

```
git checkout master
git merge <develBranch>
```

To delete the branch you've developed in:

```
git branch -d <develBranch>
```

But make sure you've committed all your changes before doing so!

Before a pull request

Before making a pull request, you want to have all the other developments that might have occurred on the `master` branch incorporated into your branch.

```
git fetch origin
git branch -vv
```

If the `master` branch needs to be updated, do:

Rebase branch to master

```
git checkout master
git pull origin
git checkout <branchname>
git rebase master
```

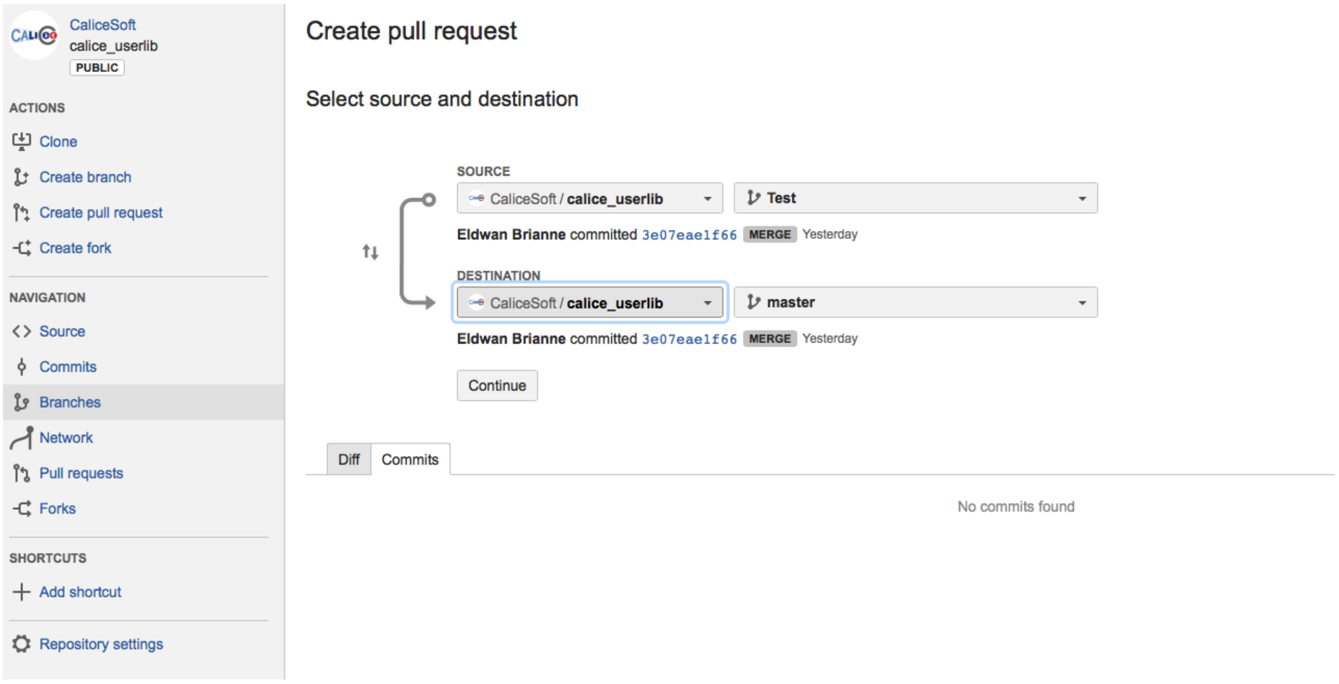
If there are any conflict, git will tell how to resolve them. Once the conflicts are resolved, you can push the branch to the `remote`.

Then you can push the local changes to your "**downstream**" remote

Push to downstream

```
git push downstream <branchname>
```

This will push the new branch `<branchname>` to the `remote` repository. Then a pull request can be done via the stash interface.



Create pull request

Select source and destination

SOURCE
CaliceSoft / calice_userlib — Test
Eldwan Brianne committed 3e07eae1f66 MERGE Yesterday

DESTINATION
CaliceSoft / calice_userlib — master
Eldwan Brianne committed 3e07eae1f66 MERGE Yesterday

No commits found