FAQ

Git

Configuration

How do I configure my full name?

```
$ git config --global user.name "Full Name"
```

How do I configure my email?

```
$ git config --global user.email "full.name@company.com"
```

How do I configure my text editor?

```
$ git config --global core.editor "vim"
```

How do I configure an HTTP(S) proxy?

```
$ git config --global http.proxy 'PROXY-HOSTNAME:PROXY-PORT'
```

Substitute `PROXY-HOSTNAME` with the hostname of the HTTP(S) proxy server, and `PROXY-PORT` with the port of the HTTP(S) proxy server. When using the credential manager on Windows, the proxy must be specified scheme first (e.g. http://).

How do I add an alias for a command?

```
$ git config --global alias.<name> '<command>'
```

For example, if you want to type `git co` instead of `git checkout` you would run:

```
$ git config --global alias.co 'checkout'
```

Do you know of any aliases that are useful?

Glad you asked! Please see the Aliases page.

Cloning

How do I clone a repository?

```
$ git clone <URL>
```

How do I clone using SSH?

Make sure you have uploaded an SSH key to your GitHub account and then run:

```
$ git clone git@github.com:path/to/repo.git
```

For example, to clone the jdk repository over SSH:

```
$ git clone git@github.com:openjdk/jdk.git
```

If you don't want to type `git@github.com` every time, add an entry to your SSH configuration.

How should I structure my local clones of repositories?

A common way of structuring local repositories is by following the domain name and path convention. For an example, see below:

```
$ tree
.
 git
    github.com
        edvbld
           jdk
           loom
        openjdk
            jdk
            loom
            valhalla
 hg
     hg.openjdk.java.net
         code-tools
            jtreg
         jdk
             jdk
```

Can I have multiple local clones of a repository?

Yes, this is supported, supply a second argument to `git clone` stating the directory name of the local repository:

```
$ git clone <URL> <DIRECTORY>
```

Branches

How do I create a local branch?

```
$ git checkout -b <NAME-OF-BRANCH>
```

If you are using Git version 2.24 or newer you can also use the command: `git switch --create <NAME-OF-BRANCH>`

How do I remove a local branch?

```
$ git branch -D <NAME-OF-BRANCH>
```

What should I name my local branches?

A common way of structuring your local branches is to name them after the issue they correspond to, for example JDK-8237566.

How do I list local branches?

```
$ git branch
  JDK-8237566
* JDK-8149128
  JDK-8077146
  master
```

The currently checked out branch has an asterisk ("*") next to it.

How do I visualize branches in a graph?

```
$ git log --format=oneline --graph --all
```

How do I switch to another branch?

```
$ git checkout <NAME-OF-OTHER-BRANCH>
```

If you are using Git version 2.24 or newer you can also use the command: `git switch <NAME-OF-OTHER-BRANCH>`

Can I have different branches checked out in different local clones of the same remote repository?

Yes, this is supported. If you prefer to have one local repository per issue you are working on, then you would have a local repository *and* a local branch per issue you are working on. For example:

```
$ git clone https://github.com/<USERNAME>/jdk JDK-8123456
$ cd JDK-8123456
$ git checkout -b JDK-8123456
```

To create the first local repository representing the work on issue JDK-8123456. To create a second local repository for JDK-8654321, run:

```
$ git clone https://github.com/<USERNAME>/jdk JDK-8654321
$ cd JDK-8654321
$ git checkout -b JDK-8654321
```

How do I push a local branch to a remote repository?

```
$ git push --set-upstream <REPOSITORY> <LOCAL-BRANCH>
```

For example, if the repository you want to push corresponds to the remote named origin and your local branch is named JDK-8123456, you would run the following command:

```
$ git push --set-upstream origin JDK-8123456
```

If you are using the Skara CLI tools then and have the branch you want to publish currently checked out, then you can just run the command `git publish`

How do I remove a remote branch?

```
$ git push --delete origin <NAME-OF-BRANCH>
```

Note: the above command assumes that the branch exists in the remote repository that origin is referring to.

How do I merge commits from another branch into the current branch?

```
$ git merge <OTHER-BRANCH>
```

How do I view commits on the current branch but exclude ones on the master branch?

```
$ git log --graph --pretty=oneline --first-parent --abbrev-commit master..HEAD
```

Commits

How do I make a commit?

First ensure that you have configured your full name, email and editor. You must then add the files that you want to be part of the commit (*even* files that are already tracked by Git):

```
$ git add path/to/file1 path/to/file/2
```

You can inspect the changes that will be part of the commit by running the following command:

```
$ git diff --staged
```

You can then create the commit by running the following command:

```
$ git commit
```

The above command will open your editor where you willl the commit message. If the commit message only is a single line then you can pass it directly to commit command: `git commit -m 'Your commit message'`.

How can I see information about a commit?

```
$ git show <COMMIT>
```

Can I update a commit with more changes?

Yes, but you will be mutating history. If you have already pushed the commit, then other people might depend on the commit you are about to mutate. If you have only pushed the commit to your personal fork and you are not currently collaborating with someone, then it is fine to mutate the history (you yourself are the only one who will be affected). If you have pushed the commit to a project repository, then it is **not** fine to mutate history, since you cause problem for other contributors.

In many cases, for example when working on a pull request, you can just make an additional commit and push it. The Skara integrate command will squash (collapse) the commits in the pull request into a single commit before integrating it, so the final result will be as if you only had made one commit. If you mutate history in an active pull request, it will make it much harder for reviewers to follow the changes.

If you still want to mutate history and update the last commit, first add the files you want to add to the commit:

```
$ git add path/to/file1 path/to/file2
```

Then run the following command to amend the `HEAD` commit:

```
$ git commit --amend --no-edit
```

GitHub

Configuration

How do I setup two-factor authentication (2FA)?

https://help.github.com/en/github/authenticating-to-github/securing-your-account-with-two-factor-authentication-2fa

How do I generate a personal access token (PAT)?

Go to https://github.com/settings/tokens and and click on "Generate new token"

How do I upload an SSH key to my GitHub account?

https://help.github.com/en/github/authenticating-to-github/adding-a-new-ssh-key-to-your-github-account

Personal Forks

What is a personal fork?

A personal fork is a copy of another repository with one major difference:

- you can use a pull request to suggest that some changes from your personal fork should be incorporated into the original repository the fork was created from

How do I create a personal fork?

- If you are using a web browser, see https://help.github.com/en/github/getting-started-with-github/fork-a-repo
- If you are using GitHub Desktop, see https://help.github.com/en/desktop/contributing-to-projects/cloning-and-forking-repositories-from-github-desktop
- If you are using GitHub CLI, see https://cli.github.com/manual/gh_repo_fork
- If you are using the Skara CLI tools, see CLI Tools#Creatingapersonalfork

How do I sync my personal fork with the original repository it was created from?

Assuming you have a local clone of your personal fork and you are using the Skara CLI tools:

```
$ git sync
```

If you also want to sync your personal fork *and* your local clone of your personal fork and you are using the Skara CLI tools you can run:

```
$ git sync --fast-forward
```

If you do **not** have the Skara CLI tools installed and you have a local clone of your personal fork, you need to the following steps:

```
$ git remote add upstream <URL-FOR-ORIGINAL-OPENJDK-REPOSITORY>
$ # for each branch you want to sync
$ git checkout <BRANCH>
$ git pull upstream <BRANCH>
$ git push origin <BRANCH>
```

Pull Requests

What is a pull request?

A *pull request* is a way to suggest that some changes from a personal fork should be incorporated into the original repository the personal fork was created from. Reviewers can comment upon and need to approve a pull request before it can be integrated. The concept of a pull request is very similar to OpenJDK's concept of "RFR" emails - both are used to suggest changes and offer a way for reviewers to provide feedback on the suggested changes.

How do I create a pull request?

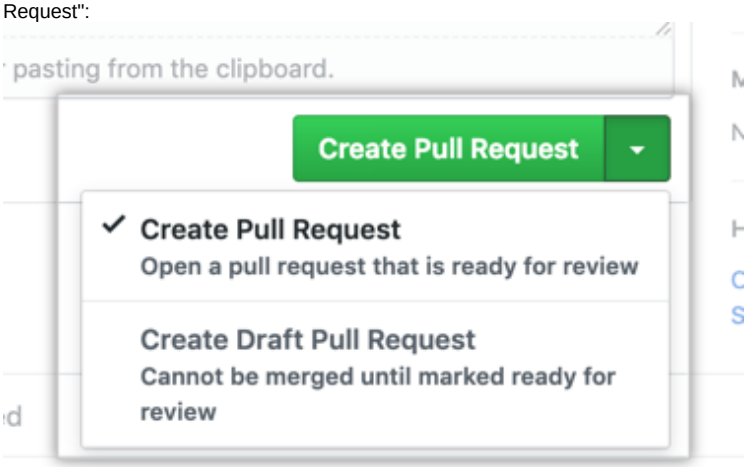
- If you are using a web browser, see https://help.github.com/en/github/collaborating-with-issues-and-pull-requests/creating-a-pull-request
- If you are using GitHub Desktop, see https://help.github.com/en/desktop/contributing-to-projects/creating-a-pull-request
- If you are using GitHub CLI, see https://cli.github.com/manual/gh_pr_create
- If you are using the Skara CLI tools, see CLI Tools#Creatingapullrequest

What is a draft pull request?

A *draft* pull request is a pull request that is not yet ready for review. Draft pull requests are primarily used to share work in a early stage or run additional testing before declaring the pull request ready.

How do I create a draft pull request?

- If you are using a web browser, click the downwards pointing arrow on the green "Create Pull Request" button and choose "Create Draft Pull Request":



- If you are using GitHub CLI, provide the `--draft` flag to `gh pr create`.
- If you are using the Skara CLI tools, provide the `--draft` to flag to `git pr create`.

How do I transition a pull request from draft to ready for review?

- If you are using a web browser, see https://help.github.com/en/github/collaborating-with-issues-and-pull-requests/changing-the-stage-of-a-pull-request
- If you are using the Skara CLI tools, run `git pr set --no-draft`.

How do I change the title of a pull request?

- If you are using a web browser, click the "Edit" button to the right of pull request's title:

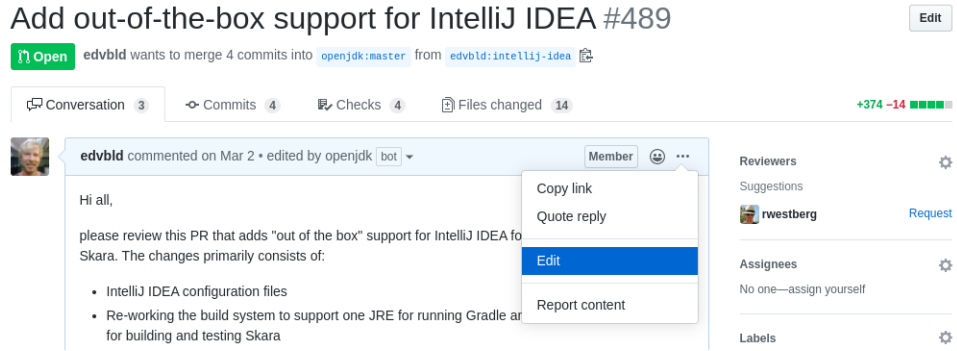
- If you are using the Skara CLI tools, run `git pr set --title='New title'`.

What is the pull request body?

The body, or description, of a pull request is the initial comment in the pull request. Can be compared to the body of the initial "RFR" email.

How do I change the body of a pull request?

- If you are using a web browser, press the three little dots in the top-right corner of the initial comment and select "Edit":



- If you are using the Skara CLI tools, run `git pr set --body`.

How do I modify the changes in a pull request?

To modify your patch in a pull request, just push more changes to the branch the pull request is based on. Avoid modifying changes that are already part of the pull request branch or force pushing unrelated changes as that will mess up the pull request and make it hard for reviewers to make sense of it. When Skara integrates the pull request, all the changes will be squashed into a single change anyway.

How do I update a pull request with upstream changes?

If your pull request slips too far behind the target branch (e.g. the master branch in the mainline jdk repository), you will need to pull in changes from master and resolve any conflicts before you can integrate. When doing so, it's usually preferred to merge the changes rather than rebasing, especially if the reviews have already started on the pull request. If changes are rebased, the history in the pull request becomes problematic and hard to follow for reviewers. When Skara integrates the pull request, all the changes will be squashed into a single change anyway.

Repositories

Can Committers for a project push directly to the project repository?

This depends on the project - some projects (typically those in a prototype phase) give project Committers direct push access, but most opt to use pull requests for contributions.

Should JDK Committers push directly to the openjdk/jdk-sandbox repository?

Yes, JDK Committers (and above) are allowed to push directly to the openjdk/jdk-sandbox repository. See the documentation for the how to name your branches to avoid conflicting with other Committers' branches.

SSH

Keys

How do I generate a private and public SSH key to use with GitHub?

```
$ ssh-keygen -t ed25519 -C github -f filename/for/the/private/key
```

The public key will be stored next to the private key and have the same as the private key but will have the suffix `.pub`.
Note: pick a strong password for your key. You will *not* have to type your password every time you want to use your SSH key if you use the SSH agent.

How do I add an SSH key to my GitHub account?

https://help.github.com/en/github/authenticating-to-github/adding-a-new-ssh-key-to-your-github-account

How do I get around having to type my password every time I want to use my SSH key?

You use the ssh-agent. An ssh-agent should already be running if you are using GNU/Linux, macOS or Windows 10 (since version 1803). To add a key to the `ssh-agent`, run `ssh-add`:

```
$ ssh-add /path/to/the/private/key
```

If you want the key to exist in the `ssh-agent` only for a limited time, use the `-t` (for timeout) option:

```
$ ssh-add -t SECONDS /path/to/the/private/key
```

How do I list keys added to the ssh-agent?

```
$ ssh-add -l
```

How do I remove a key I added to the ssh-agent?

```
$ ssh-add -d /path/to/public/key
```

Note: if the only difference in filenames between the private key and the public key is that the public key ends in `.pub`, then you can use the path to private key in the above command as well.

Configuration

GitHub SSH URLs are cumbersome to type, can I make them shorter?

Yes, by adding a host alias for github.com in your SSH configuration file `~/.ssh/config` (`C:\Program Files\Git\etc\ssh\ssh_config` on Windows). Add the following lines to the SSH configuration file:

```
Host github.com github gh
User git
Port 22
Hostname github.com
```

You can now clone from GitHub using the aliases listed after `Host`:

```
$ git clone github.com:openjdk/jdk
$ git clone github:openjdk/jdk
$ git clone gh:openjdk/jdk
```

My computer is behind an HTTP(S) proxy, can I still clone using SSH?

Yes, by adding a `ProxyCommand` directive in your SSH configuration file `~/.ssh/config` (`C:\Program Files\Git\etc\ssh\ssh_config` on Windows). The value for `ProxyCommand` is operating system dependent, since different programs are available on different operating systems. Please follow the instructions specific for your operating system below:

GNU/Linux

Add the following lines to `~/.ssh/config`:

```
Host github.com github gh
User git
Port 22
Hostname github.com
ProxyCommand nc -X connect --proxy PROXY-HOSTNAME:PROXY-PORT %h %p
```

Substitute `PROXY-HOSTNAME` with the hostname of the HTTP(S) proxy server, and `PROXY-PORT` with the port of the HTTP(S) proxy server.

macOS

Add the following lines to `~/.ssh/config`:

```
Host github.com github gh
User git
Port 22
Hostname github.com
ProxyCommand /usr/bin/nc -X connect -x PROXY-HOSTNAME:PROXY-PORT %h %p
```

Substitute `PROXY-HOSTNAME` with the hostname of the HTTP(S) proxy server, and `PROXY-PORT` with the port of the HTTP(S) proxy server. Note that the full path to `nc` must be used since Homebrew might install the GNU version of `nc` (which does not support the `-X` flag).

Windows

Add the following lines to `C:\Program Files\Git\etc\ssh\ssh_config`:

```
Host github.com github gh
User git
Port 22
Hostname github.com
ProxyCommand C:\Program Files\Git\mingw64\bin\connect.exe -H PROXY-HOSTNAME:PROXY-HOST %h %p
```

Substitute `PROXY-HOSTNAME` with the hostname of the HTTP(S) proxy server, and `PROXY-PORT` with the port of the HTTP(S) proxy server. Note that the program `connect.exe` is part of Git for Windows, if you install Git via some other mechanism then you might have to install `connect.exe` yourself.

Shell

Bash

Can I see the currently checked out local branch in my prompt?

Yes, add the following to your `~/.bashrc`:

```
git_branch() {
    BRANCH="$(git symbolic-ref --short HEAD 2>/dev/null)"
    if [ "${BRANCH}" != "" ]; then
        printf " (${BRANCH}) "
    else
        printf ""
    fi
}
PS1="${PS1}\$(git_branch)"
```

For more information on how to customize your Bash prompt, see the Arch Linux wiki for a good reference. For more Git information in your prompt, see contrib/completion/git-prompt.sh.

Can I get auto-completion for git commands?

Yes, on all of GNU/Linux, macOS and Windows this is provided out of the box when you install Git. If you want to install auto-completion manually, run the following commands:

```
$ curl -o ~/.git-completion.bash https://raw.githubusercontent.com/git/git/master/contrib/completion/git-completion.bash
$ echo 'source $HOME/.git-completion.bash' >> ~/.bashrc
```

Zsh

Can I see the currently checked out local branch in my prompt?

Yes, add the following to your `~/.zshrc`:

```
autoload -Uz vcs_info
precmd_vcs_info() { vcs_info }
precmd_functions+=( precmd_vcs_info )
setopt prompt_subst
PROMPT="$PROMPT\$vcs_info_msg_0_"
zstyle ':vcs_info:git:*' formats ' (%b) '
zstyle ':vcs_info:*' enable git
```

For more information, see the Pro Git book on Zsh.

Can I get auto-completion for git commands?

Yes, Zsh ships with auto-completion for Git out of the box. Enable auto-completion by adding the following line to `~/.zshrc`:

```
$ echo 'autoload -Uz compinit && compinit' >> ~/.zshrc
```

Editor

Emacs

Can I get Git information in Emacs?

Yes, see the package Magit.

Can I interact with GitHub from Emacs?

Yes, see the package Forge.

Vim

Can I get Git information in Vim?

Yes, see the plugin vim-fugitive. If you want to see the files that have been changed compared to `HEAD` in the "gutter" (sign column), see vim-gitgutter.

Can I interact with GitHub from Vim?

No.

Visual Studio Code

Can I get Git information in Visual Studio Code?

Yes, Visual Studio code has built-in support for Git. For details, see the Visual Studio Code documentation.

Can I interact with GitHub from Visual Studio Code?

Yes, see the plugin GitHub Pull Requests and Issues.