Git usage

Your Rating: ☆☆☆☆☆ Results: ★★★☆☆ 166 rates

I don't even know Git !

> *Git is a free & open source, distributed version control system designed to handle everything from small to very large projects with speed and efficiency.*
>
> *Every Git clone is a full-fledged repository with complete history and full revision tracking capabilities, not dependent on network access or a central server. Branching and merging are fast and easy to do.*
>
> *Git is used for version control of files, much like tools such as Mercurial, Bazaar, Subversion, CVS, Perforce, and Team Foundation Server.*

- A quick and pretty reminder of most basic commands: http://rogerdudler.github.com/git-guide/

- A free book: http://progit.org/book/ (support the authors, buy a copy!)
- The Git homepage: http://git-scm.com/
- The Git reference http://gitref.org/
- The Git community book: http://book.git-scm.com/
- If you're still up for more reading, this should quench your thirst: http://schacon.github.com/git/user-manual.html
- A StackOverflow guide/meta thread: http://stackoverflow.com/questions/315911/git-for-beginners-the-definitive-practical-guide
- An interactive cheat sheet: http://www.ndpsoftware.com/git-cheatsheet.html
- DZone refcard: http://refcardz.dzone.com/refcardz/getting-started-git
- Lars Vogel's Git tutorial: http://www.vogella.com/articles/Git/article.html
- 'Git from the bottom up' (extensive PDF): http://ftp.newartisans.com/pub/git.from.bottom.up.pdf
- ...
- Google !
  (please add your own and/or leave comments as to which of these guides/docs work the best)

Setup

Generally we recommend you use one of the latest versions of GIT.

No GIT client yet

If you have no GIT client yet, get the latest version of the official installer.

I already have GIT

```
$ git --version
git version 2.17.2 (Apple Git-113)
```

If you already have a GIT client installed, **please check** the page Supported platforms (for bitbucket) from Atlassian, and make sure that your version has been tested against our Bitbucket version. (To grab the current version of the Magnolia Bitbucket instance, look at the bottom of your profile page.)

If you have a too old version, please upgrade.

OSX: Use homebrew, macports, or simply the official installer.

On Windows: check this out:http://blog.tfnico.com/2012/04/my-git-setup-on-windows.html

On Linux: you're a big boy, aren't you? (seriously though, links/instructions welcome. Please update this page)

Configure name and username

Set `user.email` and `user.name` . ⚠ Your email *must* be the Magnolia email address!

```
git config --global user.email "john@doe.com"
git config --global user.name "John Doe"
```

✓ Execute `git config --list` to show your current settings.

Display the current branch name in your terminal

I could not work without this anymore. Add the following to your `~/.profile`:

```
# Considering you had a PS1='\h:\w \u\$ ', add this:
function parse_git_branch_and_add_brackets {
  git branch --no-color 2> /dev/null | sed -e '/^[^*]/d' -e 's/* \(.*\)/\ \[\1\]/'
}
export PS1="\h:\w \u\[\033[0;32m\]\$(parse_git_branch_and_add_brackets) \[\033[0m\]\$ "
```

Auto-completion

You won't want to use Git without this anymore. If you used the OSX installer, add this to your `/.profile`:

```
source /usr/local/git/contrib/completion/git-completion.bash
```

If you installed Git with MacPorts, then add this line to your `~/.profile`:

```
source /opt/local/share/doc/git-core/contrib/completion/git-completion.bash
```

Git at Magnolia - The following information is very much outdated. We now use BitBucket for git.

Basics

Repositories hosted at `git.magnolia-cms.com` are accessible via a number of URLs:

- read-only over HTTP: `http://git.magnolia-cms.com/<repo>.git`. You can use this to clone, fetch and pull a repository, but you won't be able to push. This is only available for our open-source projects.
- authenticated HTTPS: `https://git.magnolia-cms.com/<repo>.git`. Allows you to push - if you're allowed to for the given repository, anyway. Use your Magnolia credentials. (see #HTTP credentials below)
- SSH: `git@git.magnolia-cms.com:<repo>.git`. Allows you to push - if you're allowed to for the given repository, anyway. Uses SSH keys, so make sure we have it.
- Lastly, you can also browse them on Gitweb at `http://git.magnolia-cms.com/gitweb/?p=<repo>.git`.

... where `<repo>` is the name or path of the repository you want to clone, e.g. `git clone git@git.magnolia-cms.com:modules/dms`.

To know what repositories are available, or to validate your configuration, you can simply try to connect to the server:

```
# Anonymous:
curl -s http://git.magnolia-cms.com/info
# HTTPS:
curl -u <user>:<pass> -s https://git.magnolia-cms.com/info
# Or if you prefer to use the SSH protocol, and we have your key:
ssh git@git.magnolia-cms.com info
```

This should get you a message such as

```
hello jdoe, this is gitolite v2.3-2-gd75a165 running on git 1.7.9.4
the gitolite config gives you the following access:
     R   W         modules/foobar
     R            modules/bazqux
    @R_ @W_        testing
Please see http://wiki.magnolia-cms.com/display/DEV/Git+usage for details.
```

This means you can read/write the `modules/foobar` repository, and have read-only access to the `modules/bazqux` repository. Additionally, you have full access to the `testing` repository, like everyone else.

If on the other hand, if you get this message:

```
Permission denied (publickey).
```

... that means your public key isn't configured properly (either on your machine, or on the server). To only way to fix this at the moment is to contact me directly; send me your public key.

Repositories

In general, the `info` command (see #Basics above) or GitWeb should tell what repositories exists and are available to you.

Have a look at Git repositories structure for details on how repositories are structured.

Hey, this repository moved !?

Have a look at Git repositories structure for details on how repositories are structured, and where projects migrated from SVN have been moved to.

Hey, now I have been granted write access to this repository, what do I do ?

In both cases, what you need to do is change the URL of the "remote" you're using for that repository. If you're familiar with Git, you'll know how to do this, and you might even feel comfortable with using multiple remotes. If not, this is a simply way to change the default remote (usually named `origin`):

```
# See the current remotes and their URLs:
git remote -v
git remote set-url origin <new-url>
# Check:
git remote -v
```

So whether a repository has been moved (perhaps to a new subfolder on the same server, perhaps to another server), renamed, or you now need to use HTTPS or SSH instead of HTTP, you know what to do.

Tags, branches, detached mode

Since many of our Git repositories have been migrated from our Subversion repository (see below), there are a lot of projects where tags and branches have the same name (typically `magnolia-module-xyz-1.2` is both a branch and a tag). If you attempt to `git checkout magnolia-module-xyz-1.2`, you'll end up in detached mode, AND your local copy will reflect the contents of the *tag*. If you did want the tag, all is fine and dandy. If you wanted the branch, you need to do two things:

1. provide git with the *full* name of the reference for this branch. `origin/magnolia-module-xyz-1.2` usually does the trick, although the real full name is likely going to be `refs/remotes/origin/magnolia-module-xyz-1.2`.
2. create the branch locally and track it (if you checkout a branch, by default it's "remote", which is why you'll still be in detached mode if you just do `git checkout origin/magnolia-module-xyz-1.2`, so you'll need to instruct Git to "create" that branch locally and map it to the remote branch)
   The long version of this is

   ```
   git checkout -b magnolia-module-xyz-1.2 refs/remotes/origin/magnolia-module-xyz-1.2
   ```

   .. but a shortcut that should work just as well for more situation is:

   ```
   git checkout -t origin/magnolia-module-xyz-1.2
   ```

HTTP credentials

If you have an account at Magnolia (as a community/forge member, a customer, or even an employee😎), you should be able to clone repositories from our Git repository. If you don't, the open source repositories should still be visible on GitWeb; you should be able to clone them anonymously.

If your credentials don't seem to work like they did with SVN, please get in touch !

To avoid typing your credentials over and over again, you should **make sure you have Git 1.7.9+**, and do one of the following:

```
# On OSX:
## Check if you have osxkeychain installed
git credential-osxkeychain
## if yes then tell git to use them
git config --global credential.helper osxkeychain
## If you do not have osxkeychain helper, you can simple install it (see below "Add keychain credential support
if you use the Git installer"). Or you can use store helper instead.

# On another OS:
git config --global credential.helper store # stores credentials in ~/.git-credentials
```

There are other credential storage available depending on your installation; find them with `git help -a | grep credential-`. For more details, see http://www.manpagez.com/man/7/gitcredentials/

If you're using the Mac OSX installer, it seems the keychain credentials support isn't included yet - vote for http://code.google.com/p/git-osx-installer/issues/detail?id=82 - in the meantime, you can download and install it manually:

Add keychain credential support if you use the Git installer

```
curl -s -O http://github-media-downloads.s3.amazonaws.com/osx/git-credential-osxkeychain
chmod u+x git-credential-osxkeychain
sudo mv git-credential-osxkeychain /usr/local/git/bin/
```

Subversion migration

The server and migration of the canonical/remote repositories is described on our SYS wiki.

The migration **status** is documented on the Git repositories structure page.

To switch your **local** copy from Subversion to Git, just do a clone, and replace the directories. Since the Subversion repository remains accessible in read-only mode, you can apply any local changes you had with commands along these lines: (⚠ this script is only an **example**, take it with a pinch of salt and adapt it to your own needs)

```
cd some-project # /path/to/your/svn/working/copy
svn diff > ~/temp.patch
mv `pwd` `pwd`-svn-backup
cd ..
git clone git@git.magnolia-cms.com:some/project
cd project
patch -p 0 < ~/temp.patch
```

Gitolite

Our canonical repositories are hosted via Gitolite. As a user, you might want to read http://sitaramc.github.com/gitolite/user.html.

GitHub ?

Some of our repositories are mirrored on GitHub. See https://github.com/Magnolia/. These are copies (sometimes partial) of our canonical repositories. As such, **pushing to GitHub will not work**. Well, it will, but your changes will be overriden by the next push to the canonical repository.

User repositories

If the connection test showed you a line like `C R W user/jdoe/.*`, you have the possibility to create your own repositories.

To start a new one, just do

```
# HTTPS:
git clone https://git.magnolia-cms.com/user/jdoe/mynewrepo
# SSH:
git clone git@git.magnolia-cms.com:user/jdoe/mynewrepo
```

If you already have a local repository, you can push to this new one with

```
git remote add origin git@git.magnolia-cms.com:user/jdoe/mynewrepo
git push --all origin
git push --tags origin
```

Note that the `origin` name is completely arbitrary and up to you. You can use any descriptive name you like. `origin` is the default in git, so it's generally good practice to keep this name for the canonical, i.e "origin" repository.

To start a new repository with the **https** protocol the commands are as follows:

```
git clone https://git.magnolia-cms.com/path/to/repository

git remote add origin https://git.magnolia-cms.com/path/to/repository
```

Sharing your user repository

If you create a user repository as above and want to get other people to read or write to it, see http://gitolite.com/gitolite/g2/setperms.html

It uses `WRITERS` and `READERS`, which are arbtitrary "roles" that we added in Gitolite for user repos; usernames are LDAP UIDs. (http://gitolite.com/gitolite/g2/user.html#user_set_get_additional_permissions_for_repos_you_created_)

```
# See current perms
ssh git@git.magnolia-cms.com getperms <repo-path>

# Setup one committer
echo "WRITERS <username>" | ssh git@git.magnolia-cms.com setperms <repo-path>

# Note that the above wiped out any WRITERS and READERS that this repository had previously, except those
configured in Gitolite itself; use your bash-fu to _add_ users
```

Example:

```
echo "WRITERS mgeljic
READERS gjoseph pmundt" | ssh git@git.magnolia-cms.com setperms user/ejervidalo/amplify/events-showcase.git
```

Administration

At the moment, Pete Ryland, Philip Mundt and Grégory Joseph have administrative access to the Git server. So contact either of us to create repositories or user accounts.

For details on our setup, see the Git page in Systems & Infrastructure space.

Tips, tricks and other troubleshooting

Please see Git tips and tricks for more tips, tricks and advice !