git-resources

Git resources

web resources

tutorial

- a good french tutorial: http://www.siteduzero.com/informatique/tutoriels/gerez-vos-codes-source-avec-git
- git from bottom up (very good for a global vision): http://ftp.newartisans.com/pub/git.from.bottom.up.pdf

references

- pro-git book: http://git-scm.com/book
- a git reference that points to git man pages and pro git book chapters: http://gitref.org/
- visual git reference: http://marklodato.github.com/visual-git-guide/index-en.html
- a reference on workflow and branching model: http://nvie.com/posts/a-successful-git-branching-model/

presentation

Git introduction for north developers: git presentation ppt

Tips and tricks for git

Common errors

After a git command implying a connection to a remote repository (like git fetch), you have an error saying something like "Hung up unexpectely", if so check that :

- the remote repo is available (is git daemon launched on the remote machine ?)
- you have launched your PAgent with your key, needed to be authentified on the remote machine.

Setup your environment

Refresh automatically eclipse .project and .classpath after a branch change (checkout)

Git has a way to fire off custom scripts when certain important actions occur, here we are interested in post-checkout event.
Just put this **post-checkout** script in your <my git repo>\.git\hooks\ directory.

Activate git auto-completion of commands, branches, repos

- Copy (or use in place) this file to your home and check that it is executable

```
cp -p /usr/share/doc/git-1.7.4.1/contrib/completion/git-completion.bash
~/bin/git-completion.sh
```

- Source it in your .bash_rc or .bash_login

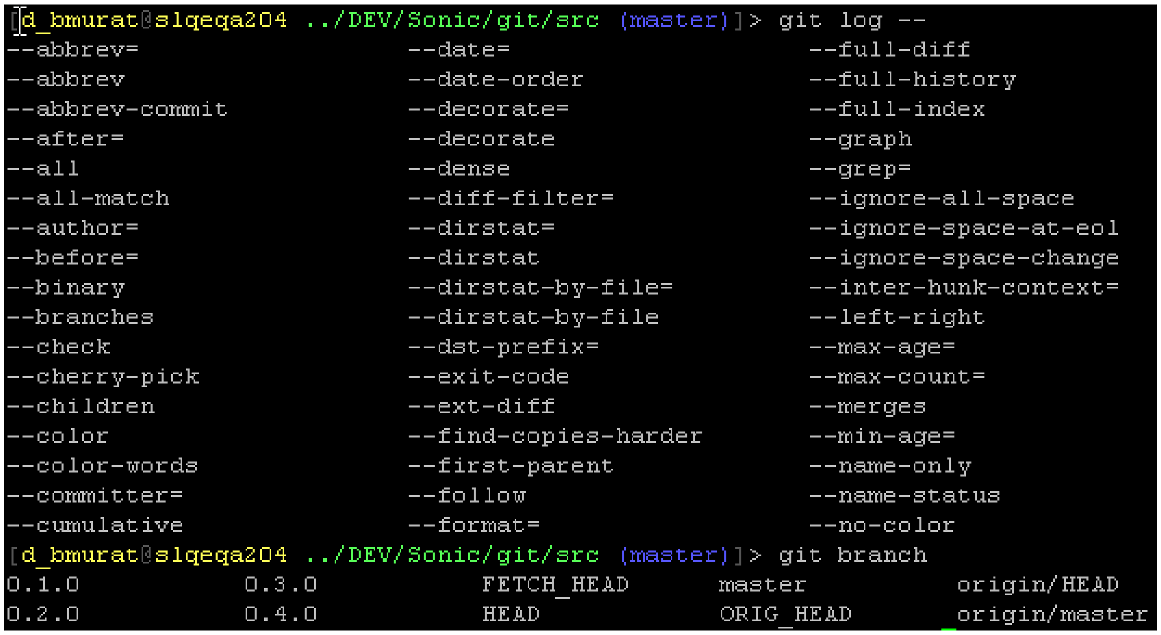
.bash_rc

```
# git
source ~/bin/git-completion.sh
```

- Source your .bash_rc

```
source .bash_rc
```

- Result

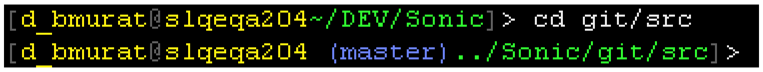


```
[d_bmurat@slqeqa204 ../DEV/Sonic/git/src (master)]> git log --
--abbrev=                      --date=                        --full-diff
--abbrev                       --date-order                   --full-history
--abbrev-commit                --decorate=                    --full-index
--after=                       --decorate                     --graph
--all                          --dense                        --grep=
--all-match                    --diff-filter=                 --ignore-all-space
--author=                      --dirstat=                     --ignore-space-at-eol
--before=                      --dirstat                      --ignore-space-change
--binary                       --dirstat-by-file=             --inter-hunk-context=
--branches                     --dirstat-by-file              --left-right
--check                        --dst-prefix=                  --max-age=
--cherry-pick                  --exit-code                    --max-count=
--children                     --ext-diff                     --merges
--color                        --find-copies-harder           --min-age=
--color-words                  --first-parent                 --name-only
--committer=                   --follow                       --name-status
--cumulative                   --format=                      --no-color
[d_bmurat@slqeqa204 ../DEV/Sonic/git/src (master)]> git branch
0.1.0           0.3.0           FETCH_HEAD      master          origin/HEAD
0.2.0           0.4.0           HEAD            ORIG_HEAD       origin/master
```

Add current branch info in linux prompt (needs git completion)

- Add the `__git_ps1`variable to your usual PS1 variable

```
PS1="...$(__git_ps1)..."
```

- Result


```
[d_bmurat@slqeqa204~/DEV/Sonic]> cd git/src
[d_bmurat@slqeqa204 (master)../Sonic/git/src]>
```

Activate colors in git output messages on linux

```
git config --global color.ui true
```

Pretty format the output of git log command

Add the global setting to always log with the pretty format:

```
git config --global format.pretty '%Cred%h%Creset -%C(yellow)%d%Creset %s
%Cgreen(%cr) %C(bold blue)<%an>%Creset'
```

Create an alias `lg` with useful parameters:

```
git config --global alias.lg "log --graph --abbrev-commit --date=relative"
```

to use:

```
git lg
```

Add aliases

Find your git global settings: `.gitconfig` :

- use tortoise git `-> settings`
- select `Git` in left menu and click on "`Edit global .gitconfig`"

and add the followiing aliases:

```
[alias]
    st = status -s
    ci = commit
    cia = commit --amend
    co = checkout
    br = branch
    sb = show-branch
    cp = cherry-pick
    staged = diff --cached
    rb = rebase
    rbc = rebase --continue
    rbs = rebase --skip
    rl = reflog
    rs = remote show
    rt = remote
    ru = remote update
    rp = remote prune
    sm = submodule
```

Useful commands

- show information about remote repo (url, tracked branches)

```
git remote show origin
```

- Create remote branch

```
# create local branch
git checkout -b <branch_name>

# create remote branch that will be tracked by local branch
git push -u origin <branch_name>

# check that everything is ok
git branch -avv
```

- Remove remote branch

```
# you need to be on another branch that the branch you want to remove
# remove remote branch (ok, it's weird but you push nothing - the nothing
before ':' - to remote on ref <branch name>)
# git push [remotename] [localbranch]:[remotebranch]
git push origin :<branch name>

# or simpler
git push origin --delete <branch name>

# now you have to delete your local branch
git branch -d <branch name>

# check that everything is ok
git branch -avv
```

- Show locally modified files:

```
# modified files but not added in index
git diff --stat

# modified files but added in index (diff with last commit)
git diff HEAD --stat

# show modifications on a specific file
git diff <path/file>
or
git diff HEAD <path/file>
```

- Add files to git index (to prepare a commit)

```
# add specifics files
git add <file1> <file2> <...> <fileN>

# add all modified and new files to index
git add .
```

- Show all files added to index (files that will be commited)

```
git diff --stat --cached
```

- Commit files that have been added to index

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

# sample:
git ci -m "fixes NMAIAMAPI-709: Refactor process that forces the seqnum (due
to change in qfj-1.5.3)"
```

- Rollback last commit

```
git reset --soft HEAD~1
```

- Fetch remote commit (it is better to use fetch/rebase than pull)

```
git fetch
```

- Move local commit after last commit retrieved from remote repo

```
git rebase
```

- Modify local commits history before pushing to remote repo (merge, remove commit, change commits order)

```
git rebase -i origin/master
```

- Resolve merge conflict status after resolution

```
git add <mon fichier résolu> <mon 2e fichier résolu>
```

resume the rebase

```
git rebase --continue
```

stop rebase and come back to state before rebasing

```
git rebase --abort
```

- Push local commits to remote repo (don't modify history after push)

```
git push
```

- record the current state of the working directory and the index and go back to a clean working directory to work on another issue

```
# anonymous stash (not recommended)
git stash

# named stash
git stash save "<save state name>"
```

- List all stashs

```
git stash list
```

- Apply last stash on local directory (fifo order) and remove it from the list

```
git stash pop
```

- Apply a specific stash

```
git stash apply <index from stash list>

# remove applied stash
git stash drop <index de la stash list>
```

- Remove a tag

```
# remove local tag
git tag -d <TAGID>
# push removed tag on remote repo
git push origin :refs/tags/<TAGID> (or much clearer : git push --delete origin
<TAGID>)
```

- Rename a tag

```
git tag <NOUVEAU-TAG> <ANCIEN-TAG>
git tag -d <ANCIEN-TAG>
git push --tags
git push origin :refs/tags/<ANCIEN-TAG>
```

- Cherry-picking n-1 commit of a specific branch without auto commit

```
git cherry-pick --edit --no-commit -x 1.2.x~1
```

- Restore a file removed in a commit (there are other ways to do that)

```
# retrieve the commit where the file have been removed
# (in fact the last commit where the path object have been modified)
git rev-list -n 1 HEAD -- <filepath>

# display commit details
git show <commit sha1>

# checkout file version of preceding commit (before deleted)
git checkout <commit sha1>^ -- <filepath>
```