Git Cheat Sheet

What I just learned about Git

Settings

- list Git configuration:

```
git config --list
```

- set configuration:

```
git config --global --add user.name "John Doe"
git config --add color.ui "auto"
```

`--global` makes the setting global for all repos.

- show configuration:

```
git config --get user.name
```

- some common settings:

name	meaning
user.name	User name.
user.email	User email.
color.ui	Use colors ("auto"!)?
core.editor	Which editor to use?

Basic Git

- Create a repository:

```
cd projDir
git init
```

- Add files to repository:

```
cd projDir
git add file1 file2
```

or

```
git add *
```

- Status and log:

```
git status
git log
```

- Commit changes:

```
git commit changedFile -m "Commit message."
```

or

```
git commit -a
```

(opens editor and allows to change the commit message)

- remove file from version control:
    - also remove file from disk:

```
git rm file
```

  - keep file on disk:

```
git rm --cached file
```

- go back to `fileName`'s last commited version:

```
git checkout -- fileName
```

- get help:

```
git stash --help
```

  shows the man page for `git stash`.

- rename a versioned file:

```
git mv oldName newName
```

Branches

- list branches:

```
git branch
```

  Add `-r` for remote branches, use `-a` for remote and local branches.

- create new branch:

```
git branch newBranch
```

  Create and change to that branch:

```
git branch -b newBranch
```

- change to a branch:

```
git checkout branchName
```

- delete branch:

```
git branch -d branchName
```

  for branches that branch off `HEAD`,

```
git branch -D branchName
```

  for any branch.

- merge current branch with `other` branch:

```
git merge other
```

Usage with central repository

- Create a repository on central server:

```
git init --bare --shared foo.git
chgrp -R dev foo.git  (optional)
```

  `shared` makes the repo group writable.

- push local repo to server:

```
cd localRepo
git push ssh://user@host.domain.tld/home/user/foo.git '*:*'
```

(this pushes the local repo with everything to the server)

- after hacking, update repo on server:

```
cd localRepo
git push
```

GitHub

- create repository `repoName` from the web interface
- teach local repository about the remote one:

```
cd repoName
git remote add origin git@github.com:githubuser/repoName.git
```

- push files to GitHub:

```
cd repoName
git push
```

- to clone the GitHub repo:

```
git clone git@github.com:githubuser/repoName.git newRepo
```

- push changes back to GitHub:

```
cd repoName
git push
```

Going back to checked in versions

There are at least two different ways to reset to working directory to the last versioned status:

Checkout: Forget about changes

```
git checkout -- fileName
```

resets `fileName` to the last checked in version.

Stashes: keep changes

- changes in a working dirctory may be 'stashed' away:

```
git stash save "Status before going back"
```

- stashes are listed with:

```
git stash list
```

- apply the stash on top of the stack again:

```
git stash apply
```

keeps to stash saved, whereas

```
git stash pop
```

applies the stash and also removes the stash form the list.

- delete a stash:

```
git stash drop
```

deletes the stash on top of the stack, whereas

```
git stash drop stash@{0}
```

deletes the stash `stash@{0}`.

Links

- Git reference: http://gitref.org/
- Git community book: http://book.git-scm.com/
- Git with central sever: http://toroid.org/ams/git-central-repo-howto

TODO

- notions (staging, head...)
- learn about branches