

Join GitHub today

GitHub is home to over 36 million developers working together to host and review code, manage projects, and build software together.

Sign up

Dismiss

A collection of cheat sheets usefull for web development <https://groupe-sii.github.io/cheat-sh...>

355 commits

3 branches

0 releases

16 contributors

Branch: develop

New pull request

Find File

Clone or download

vogloblinsky

Merge pull request #40 from amalricBzh/develop ...

Latest commit 90125b0 on Mar 22

assets/images	Merge pull request #38 from mrgngautier/agile	6 months ago
src	feat(doc): Git cheatsheet	4 months ago
.editorconfig	chore(.editorconfig): add .editorconfig	2 years ago
.eslintrc	add eslint	3 years ago
.gitignore	Fix #22 by removing hard coded fixed version of dependencies	last year
.travis.yml	chore(trzvis): remove scure token from travis.yml	2 years ago
README.md	add AGILE category	7 months ago
deploy.js	fix Merge from LoicC04	8 months ago
gulpfile.js	feat(doc): Stencil cheatsheet started	6 months ago
package.json	Merge pull request #36 from Madikera77/develop	6 months ago

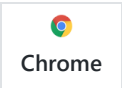
README.md

Cheat Sheet Generator

Hosted on github pages

Cheat sheets are hosted on github pages : <https://groupe-sii.github.io/cheat-sheets/>

Browsers support made by godban



| last 2 versions

Install

NodeJS version 6+

gulp is needed in global in order to run compilation :

npm install gulp -g

yarn OR npm install

Usage

From install folder:

```
gulp create-new-cheat-sheet --name <name> --category <tools|frameworks|languages|gile>
```

Put your svg/png logo in assets/images folder Put your commands or codes on:

- src/<name>/first-side/column1.md
- src/<name>/first-side/column2.md
- src/<name>/reverse/column1.md
- src/<name>/reverse/column2.md

Devtools

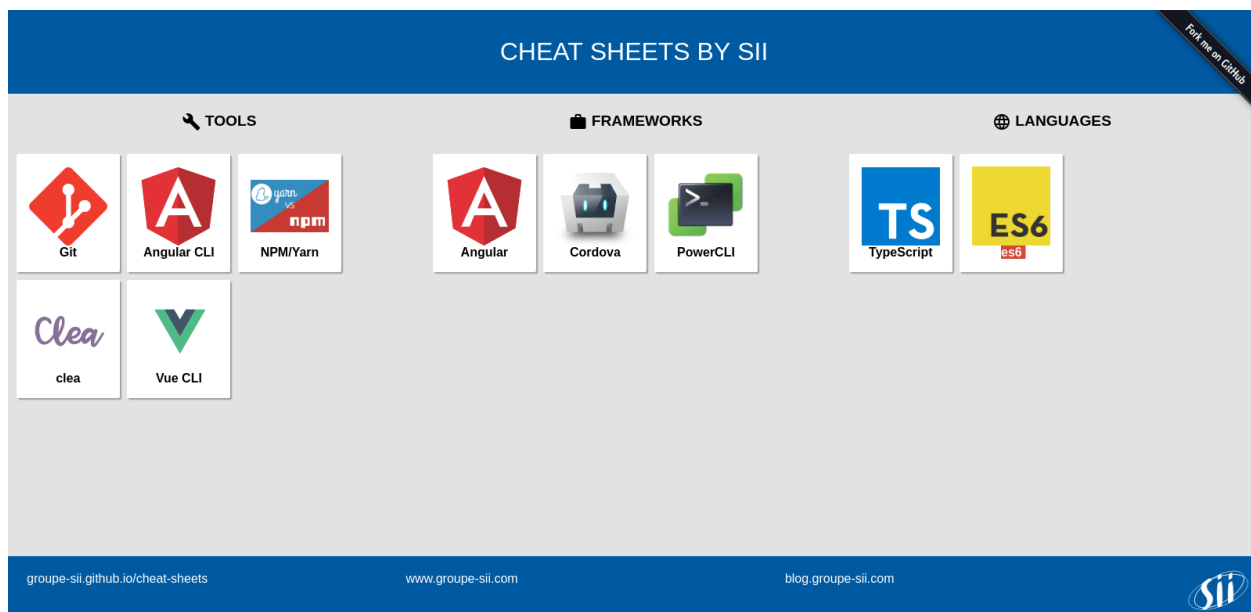
Build and reload server:



```
gulp watch
```

Print



- Hit `Ctrl+P` to generate the PDF version, using `Save as PDF`
- Disable margins

Screenshots



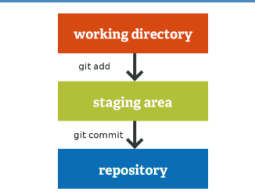



GIT CHEAT SHEET
 Improve your Git skills

git version: 2.9+ - Date: March 2017

LOCAL GIT STATE AND CHANGES

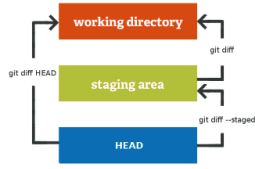


```

graph TD
    WD[working directory] -- "git add" --> SA[staging area]
    SA -- "git commit" --> R[repository]
  
```

MANAGE LOCAL CHANGES

Show changes between commits, commit and working tree



```

graph TD
    WD[working directory]
    SA[staging area]
    H[HEAD]
    WD -- "git diff HEAD" --> H
    SA -- "git diff" --> WD
    H -- "git diff --staged" --> SA
  
```

Show the working tree status

```
git status
```

Add contents to the index

```
git add <file>
```

Add current contents of the index in a new commit

```
git commit -m 'subject' -m 'body'
```

Who changed what in

```
git blame <file>
```

REMOTE

List all remotes repositories

```
git remote -v
```

Show information about remote

```
git remote show <remote>
```

Add new remote repository

```
git remote add <remote> <url>
```

Download all changes from

```
git fetch <remote>
```

Download changes and integrate into HEAD

```
git pull <remote> <branch>
```

Publish changes on a remote

```
git push <remote> <branch>
```

BRANCH

Create a new local branch and switch HEAD branch

```
git checkout -b newBranch
```

Delete a local branch

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

COMMIT MESSAGES

- Limit the subject line to 50 characters
- Do not end the subject with a period
- Use the imperative mood in the subject line
- User the body to explain *what*, *why* and *how*

Tips: Separate subject from body with a blank line. From de command line you can use:

```
git commit -m 'subject' -m 'body'
```

TAG

Tag a commit

```
git tag <tag-name>
```

MERGE

Merge on your current branch

```
git merge <branch>
```

Reapply commits (of the) on top of you current branch

```
git rebase <branch>
```

UNDO

List all operations on repository

```
git reflog
```

Discard all local changes in your working directory

```
git reset --hard HEAD
```

Discard local changes in a file

```
git checkout HEAD <file>
```

Revert a commit (add a new commit with contrary changes)

```
git revert <commit>
```

Change the last commit

```
git commit --amend
```

Be careful: don't amend a published commit!

SETTINGS

Somes commands to change the default git behavior

```
git config --global pull.rebase true
```

```
git config --global rerere.enabled 1
```

PROXY

Add proxy on your global configuration

```
git config --global http.proxy http://proxy.com:1234
```

Unset the global proxy

```
git config --global --unset http.proxy
```

Add proxy on your current project directory

```
git config http.proxy http://proxy.com:1234
```

Check your global proxy

```
git config --global --get http.proxy
```

Check your local proxy

```
git config --get http.proxy
```

COMMIT RULES

- One commit for one task: if you have 2 tasks written in one file you can use *git add -p* and chose which chunk you need to add ad your commit
- Try to commit as often as possible
- Do not commit a half-done work. You should only commit code when it's completed.
- Commit a stable version of your work: the tests should not failed

DEFAULT CLIENT

Visualize git tree

```
gitk --all
```


Graphical git operations



```
git gui
```

groupe-sii.github.io/cheat-sheets



www.groupe-sii.com

blog.groupe-sii.com







GIT CHEAT SHEET
 Improve your Git skills

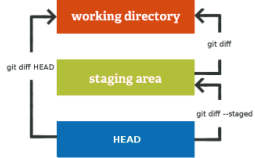
git version: 2.9+ - Date: March 2017

LOCAL GIT STATE AND CHANGES



MANAGE LOCAL CHANGES

Show changes between commits, commit and working tree



Show the working tree status

```
git status
```

Add contents to the index

```
git add <file>
```

Add current contents of the index in a new commit

```
git commit -m 'subject' -m 'body'
```

Who changed what in

```
git blame <file>
```

REMOTE

List all remotes repositories

```
git remote -v
```

Show information about remote

```
git remote show <remote>
```

Add new remote repository

```
git remote add <remote> <url>
```

Download all changes from

```
git fetch <remote>
```

Download changes and integrate into HEAD

```
git pull <remote> <branch>
```

Publish changes on a remote

```
git push <remote> <branch>
```

BRANCH

Create a new local branch and switch HEAD branch

```
git checkout -b newBranch
```

Delete a local branch


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


COMMIT MESSAGES

- Limit the subject line to 50 characters
- Do not end the subject with a period
- Use the imperative mood in the subject line
- Use the body to explain *what*, *why* and *how*

Tips: Separate subject from body with a blank line. From the command line you can use:

```
git commit -m 'subject' -m 'body'
```

groupe-sii.github.io/cheat-sheets
www.groupe-sii.com
blog.groupe-sii.com


GIT CHEAT SHEET


TAG

Tag a commit

```
git tag <tag-name>
```

MERGE

Merge on your current branch

```
git merge <branch>
```

Reapply commits (of the) on top of your current branch

```
git rebase <branch>
```

UNDO

List all operations on repository

```
git reflog
```

Discard all local changes in your working directory

```
git reset --hard HEAD
```

Discard local changes in a file

```
git checkout HEAD <file>
```

Revert a commit (add a new commit with contrary changes)

```
git revert <commit>
```

Change the last commit

```
git commit --amend
```

Be careful: don't amend a published commit!

SETTINGS

Some commands to change the default git behavior

```
git config --global pull.rebase true
```

```
git config --global rerere.enabled 1
```

PROXY

Add proxy on your global configuration

```
git config --global http.proxy http://proxy.com:1234
```

Unset the global proxy

```
git config --global --unset http.proxy
```

Add proxy on your current project directory

```
git config http.proxy http://proxy.com:1234
```

Check your global proxy

```
git config --global --get http.proxy
```

Check your local proxy

```
git config --get http.proxy
```

COMMIT RULES

- One commit for one task: If you have 2 tasks written in one file you can use *git add -p* and choose which chunk you need to add to your commit
- Try to commit as often as possible
- Do not commit a half-done work. You should only commit code when it's completed.
- Commit a stable version of your work: the tests should not failed

DEFAULT CLIENT

Visualize git tree

```
gitk --all
```

Graphical git operations

```
git gui
```

groupe-sii.github.io/cheat-sheets
www.groupe-sii.com
blog.groupe-sii.com
