Git Cheat Sheet

Required text in angle brackets: `<required>`
Optional text in square brackets: `[optional]`
Comments in parenthesis: `(comment)`

Clone a Repository (use the period to copy in the current dir instead of a new subdir)

```
git clone <url> [.]
```

Create a Repository from a Current Project

1) `git init`
2) `git add *`
3) `git commit –m "First commit"`

Update a Local Repository from a Remote

```
git pull origin <branch>
```

Push a Local Repository from a Remote

To push current branch: `git push`
To push a specific branch: `git push –u origin <branch-name>`
To push all branches: `git push --all [remote name(origin)]`
To push and force changes: `git push origin master –f` (can lose remote history!)

Checkout

```
git checkout <branch or hash> [-f (clobber non-committed changes)]
git checkout <branch or hash> <file path> (to get a single file, don't need path if in directory)
```

Add an Alias (for example show a graph using `git gr`)

Add a local alias: `git config --local alias.gr log`
Add a global alias: `git config --global alias.gr log`
List aliases: `git config --get-regexp alias`

Branch

Create and switch to new branch: `git checkout -b <branch name>`
Delete a branch: `git branch –d <branch name>`
List branches: `git branch [--all]`

Merge

Merge branch to master:

1) `git checkout master`
2) `git merge <branch>`

Merge remote branch to local:

1) `git checkout <desired branch to merge to>`
2) `git remote add <remote name> <url>`
3) `git fetch <remote name> <branch>`
4) `git merge <remote name>/<remote branch name>`

-- OR --

`1) git pull <remote name> <remote branch>` (fetch and merge)

Two files:

```
1) git merge-file [--ours] <into file> <empty file> <from file>
```

If there are merge conflicts:

```
1) Edit files shown by git
2) git add <file(s)>
3) Preview: git diff <source_branch> <target_branch>
4) Git merge <branch>
5) git commit –m "<commit message>"
```

Merge two repositories (rep_b to rep_a)

```
cd <rep_a path>
git remote add rep_b <rep_b path>
git fetch rep_b --tags
git merge --allow-unrelated-histories rep_b/master
git remote remove rep_b
```

Overwrite a single file from a different branch

```
git checkout <branch> <file path>
```

Use Notepad++ as the default Git editor

```
git config --global core.editor "'C:/Program Files
(x86)/Notepad++/notepad++.exe' -multiInst -notabbar -nosession -noPlugin"
```

Edit the configuration file (.gitconfig)

```
git config --global –edit
git config --local –edit
```

Stash (temporarily save/restore current work). Make sure files to stash are staged.

1) `git stash`
2) `(switch back to branch where stash was created from)`
3) `git stash pop` (apply stash and delete from stack)

```
git stash drop [name]       (delete stash from stack)
git checkout stash -- .     (overwrite all files)
git stash clear             (delete the stash)
```

Connect a Repository to a Remote Server

```
To add:      git remote add origin <url>
             git remote add <name> <url>
To remove:   git remote remove origin
To list:     git remote –v
To rename:   git remote rename <old> <new>
```

Compare Local Repository to Remote

```
git diff <masterbranch_path> <remotebranch_path>
git diff master origin/master
```

Remove files from staging

```
git reset HEAD -- <file(s) or directory>
```

Discard working directory changes

```
git checkout -- <file(s) or directory>
```

Discard all Local Changes Permanently

```
git reset --hard
```

After changing .gitignore

```
git rm -r --cached . or git rm --cached <file>
git add .
git commit -m "fixed untracked files"
```

Delete commit at HEAD (Danger! Whacks working directory files)

```
git reset --hard HEAD~1
```

Delete commit Locally and on Remote Repository (danger!)

```
git reset --hard <local hash>
git push --force <remote> <branch>
```

Change the current commit message (opens text editor)

```
git commit -amend
```

List all files in a branch or commit

```
git ls-tree -r --name-only <branch or hash>
```

Proxy Setup

Fdgdfg

Log formats

git config --global alias.gr "log --graph --pretty=format:'%C(yellow)%h%Creset%C(cyan)%C(bold)%d%Creset %C(cyan)(%cD)%Creset %C(green)%ce%Creset %s --all'"

Glossary

HEAD	symbolic name for currently checked out commit
DETACHED HEAD	when a hash is checked out instead of a branch