# Using GitHub

Antonio Almagro, Igor Arambasic

# Git

Do we all have a Github account?

Creating a version control repository/directory:

- @remote directory (github)
- @local directory

Git basic commands:

- clone - Clone a repository into a new directory
- init - Create an empty Git repository or reinitialize an existing one
- status - Show the working tree status
- diff - Show changes between commits
- add - Add file contents to the working tree and the index
- rm - Remove files from the working tree and from the index
- commit - Record changes to the repository
- fetch – retrieves objects and their metadata from the remote repo
- push - Update remote refs along with associated objects
- pull – fetch + merge  changes

# Git

## Creating  local repository by cloning an empty from remote repo

Lets try it (go to ~Repositories):

- Prepare the repository at github webpage (with file inside)
- git clone https://github.com/IgorAramb/master3_first.git  (link C/P from webpage)
  cd master3_first

we see master in green. What's that?

It tells us we are on master branch…
Master is a default branch name…

By default each new clone maintains a link back to its parent repository via a remote called origin!
If you don't want to use it:
git clone --origin github https://github.com/IgorAramb/master3_first.git master3_another_dir

# Git

## What is a branch?

Branches are like "Save as..." on a directory.

You can say they are as "virtual directories" in the .git folder.

# Creating  local repository by cloning an empty from remote repo

Moving on….

kwrite readme.txt
git add readme.txt

we see 2 numbers and the master is not green!
Why? What's happen?

We have un-staged changes…
first number corresponds to the number of added lines wrt last committed version
second  number corresponds to the number of delete lines wrt last committed version
git diff

# Creating  local repository by cloning an empty from remote repo

Moving on….

git commit

Commit is used to record changes to repository.
We always have to describe what has been changed so a log file opens automatically…

master has changed its color in prompt again !
Why? What's happen?

We have committed changes ready to be pushed to repository

We see only one number and it corresponds to the number of committed changes ready to be pushed

git status

# Creating  local repository by cloning an empty from remote repo

Moving on….

```
kwrite readme.txt
git add readme.txt
```

we see 3 numbers now and the master is not green again!

# Creating  local repository by cloning an empty from remote repo

Moving on….

git push -u origin master (push to which remote what branch)

-u, --set-upstream
For every branch that is up to date or successfully pushed, add upstream (tracking) reference,

master in prompt in green again!

What is origin/master? Where did we push?
git remote –v
git remote show origin

- See the repository at github webpage

# Initializing local repository and pushing it to remote repo

Lets try this(go to ~Repositories):

```
mkdir master3_second; cd master3_second
echo “my name is not important" >> README.md
git init
git add README.md
git commit -m "first commit“
```

Now create new (empty) repository @github web page

```
git remote add origin https://github.com/Your_Account_name/Your_repository_name.git
git push -u origin master
```

# Using command line for Initializing empty repo at remote and cloning it

curl -u 'USER' https://api.github.com/user/repos -d '{"name":"REPO"}'
git clone https://github.com/USER/REPO.git
Lets try it (go to ~Repositories):

- curl -u 'IgorAramb' https://api.github.com/user/repos -d '{"name":"four"}'
  git clone https://github.com/IgorAramb/four.git
  cd four
  git status
  kwrite readme.txt
  git status
  git add readme.txt
  git status
  git rm --cached readme.txt
  git status
  git add readme.txt
  git push -u origin master
  git status

# Git

alias gs="git status“

…and many more☺

# Git

File classifications in Git

- Tracked – any file already in repo or staged for commit
- Ignored – file explicitly declared as invisible or ignored inside .gitignore file
- Untracked – any file not found in previous two categories

git ls-files – show all files being tracked

cat .gitignore

Let’s make .gitignore …. And commit it!

echo "*~" >.gitignore

- emacs with git and backup files

# Git

Useful commands

- git add .  DON'T DO THIS IN YOUR HOME DIRECTORY!!!
  - Adds all the files inside the local repository and stages them for commit.
- git log - Show commit logs =(git log HEAD)
- git log + TAB

But  WAIT… why do we see Israel Herraiz as Author???

git config -l –  list the settings to all variables

Thinking as Hacker… maybe there is some git configuration file somewhere… lets find it..

How? …. use find command ☺

# Git

Change the content of

- ~/.gitconfig
- git config --global user.name "Igor A"
- git config --global user.email “ia@some.com"

Without global flag it would set repository specific information

- git config --unset user.name
- cat .git/config

Look inside the

- ~/.git-credentials -> I will NOT show this file on the screen!!!

# Git

## What is a branch?

Branches are like "Save as..." on a directory.

You can say they are as "virtual directories" in the .git folder.

But…

- common files only stored once -> no wasted space
- so you can process your work with multiple possibilities while keeping the original safe.

While inside a physical directory, you move through virtual directories with a checkout.

# Git

Let’s make some branches

- git branch bug/1
- git checkout -b bug/2
- git checkout +TAB
- git branch -m bug/1 bug/5
- git branch -d bug/5

… edit a file and commit…

git push -u origin bug2

git push origin :bug2 (delete the branch from remote)

# Git

## Merge vs Rebase

- git branch bug/10
- git branch bug/11

… edit a file in master and commit…

… do smth in branch bug/10 and and commit…

… do smth in branch bug/11and and commit…

- git checkout bug/10
- git log
- git merge master
- git log

… do the same for bug11 but with rebase

# Dealing with “oh… sh*t… wait… too late”

- Stage for commit but not committed:
  - git add readme.txt
  - git rm --cached readme.txt

  or
  - git checkout readme.txt

  or
  - git reset HEAD readme.txt

- What is HEAD?
  - A pointer to the most recent commit on the current branch
  - HEAD^ = commit -1 (parent of the most recent commit)
  - HEAD^^= commit -2
  - HEAD~N=commit -N

# Dealing with “oh… sh*t… wait… too late”

- Committed but not pushed:
  - git add readme.txt
  - git commit -m "readme.txt with new text"
  - git reset origin/master

  or
  - git reset --hard origin/master

hard = Any changes to tracked files in the working tree since <commit> are discarded.

# Dealing with “oh… sh*t… wait… too late”

- “Accidentally” deleted a file
  - rm readme.txt
  - git checkout readme.txt
- I don’t want this to be local git repository
  - rm -rf ./.git
- I have a typo in repository name
  - Change the repo name at webpage
  - git remote set-url origin https://github.com/user/NEW_NAME.git

  or
  - git remote rm origin
  - git remote add origin https://github.com/user/NEW_NAME.git

# Dealing with “what if…”

What if you want to change the name of the file in the repo?
What if you want to delete a file in the repo?
What if you want to delate a remote repo?

Use git commands:

- mv - Move or rename a file or a directory
  - Git doesn’t keep track of rename!!!
- rm - Remove files from the working tree and from the index

Don’t forget to push the changes!!

Try:

```
git mv old_name new_name
git rm file_name
```

# Dealing with “what if…”

If files a, b and c are changed, you SHOULD commit them separately!

However, we rarely do this so…

- git add -u = stage all tracked, modified files
- git commit -a -m “commiting all tracked, modified files”

# Dealing with “what if…”

Who is responsible for the error?

- git diff 2a5dbeec51426dcf77b57a18698f4a6d768d8824 8a2e6914fc8ecbe22ca4aa70c6cbd59dd098a9eb fileone2.txt
- git diff --cached
- git blame -L 5, 10 filename.txt = show who has changed lines 5 to 10 in this file
- git reset --hard HEAD
- git reset --hard aabe405e70b78908f3d5e2bbd21b3f247332036e