Josep Carreras
LEUKAEMIA
Research Institute

Introduction to Git and Github

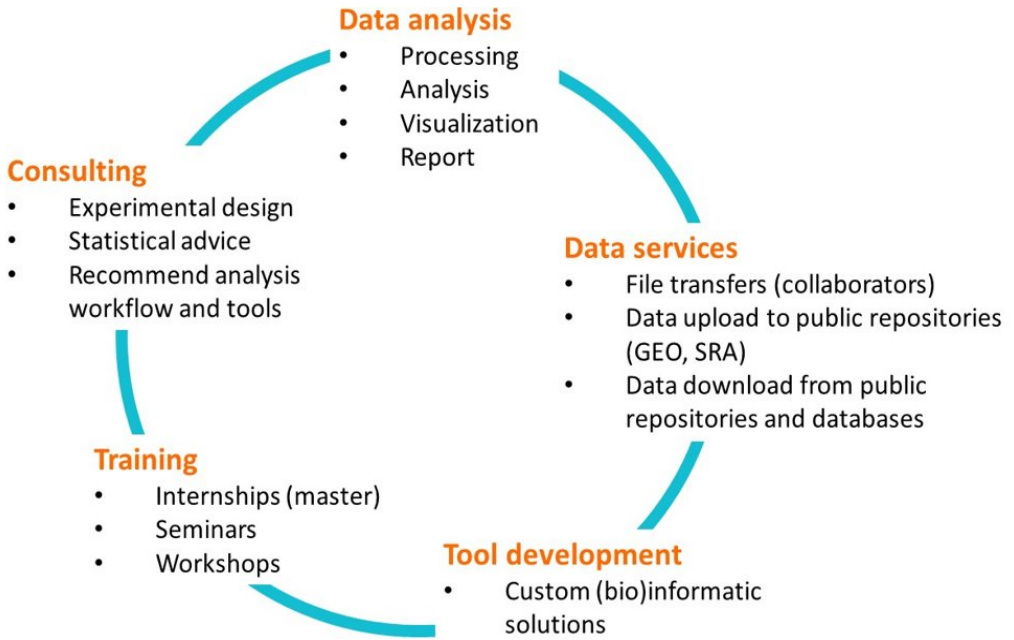
The IJC Bioinformatics Unit

Angelika Merkel
Unit Manager
Bioinformatician

Emilio Lario
Software
Engineer

Marina Vilardell
Bioinformatician

Marta Meroño
Bioinformatician
(Student)



Office: 1st floor; phone: 4300
https://ijcbit.eu
https://www.carrerasresearch.org/en/bioinformatics-unit

Josep Carreras LEUKAEMIA Research Institute

Outline

- Version Control (Git)
- Basic Git Workflow and commands
- Backtracking
- Remote repositories
- Maintaining a good repository
- Branches (optional)

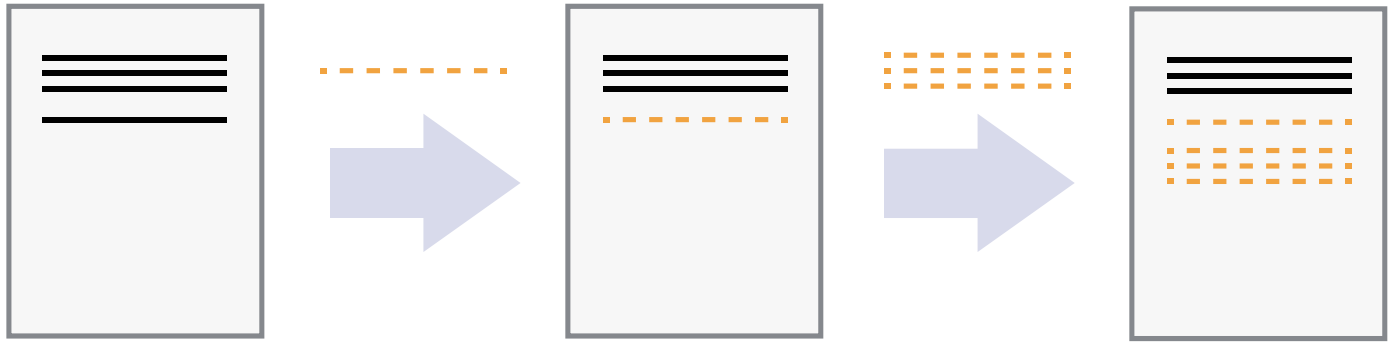
Josep Carreras
LEUKAEMIA
Research Institute

1. Version Control

Version Control system

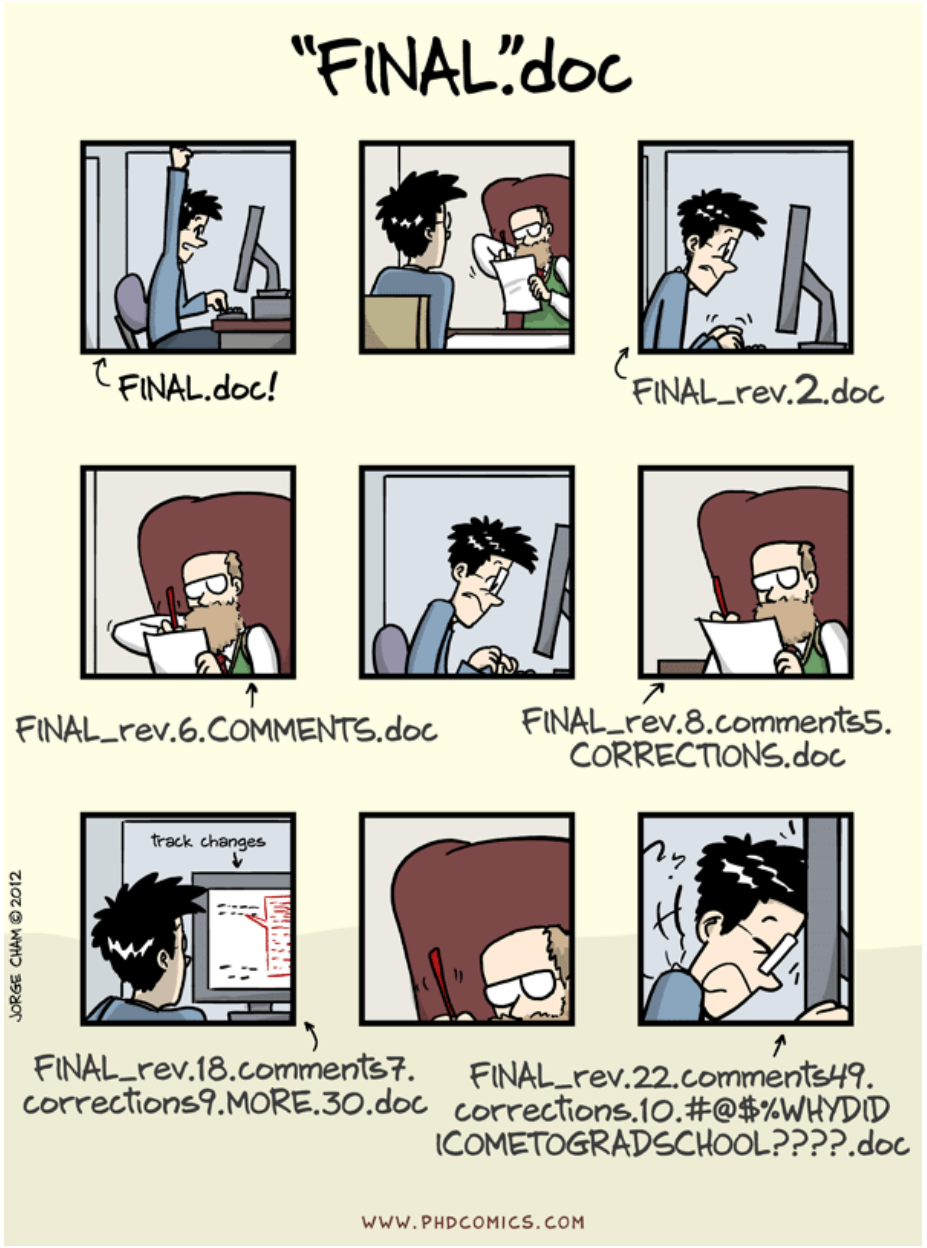
Josep Carreras LEUKAEMIA Research Institute

A version control system is used to keep track of the changes that you make to the files of a directory in your computer.



The most widely used software in the world that effectively tracks the progress of your projects over time is **Git**.

Benefits of using a version control system



- Clear and well organization of a project.
- Collaboration. Share writing and code, and to work together with multiples collaborators.
- Understanding. Version control helps to understand who wrote or contributed to parts of a project.
- Undo a set of changes, keep different versions of the same project, compare changes over time and much more!
- Backup. Your code and writing can be stored on multiple other computers.

GIT vs. GITHUB

Josep Carreras LEUKAEMIA Research Institute

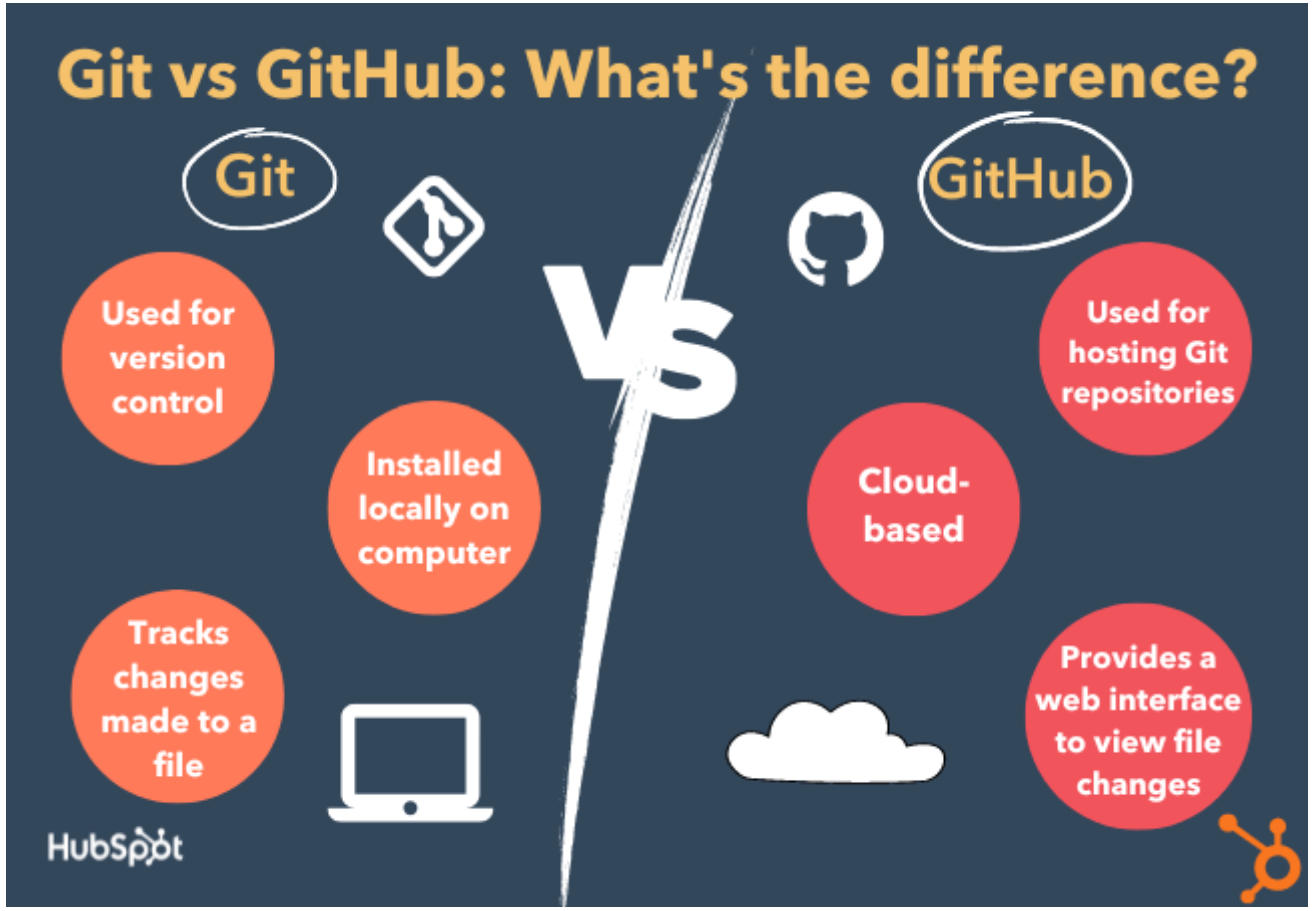
> ⚠ Warning
>
> We often hear the terms **Git** and **GitHub** used interchangeably but they are slightly different things.

GitHub is a popular website for hosting and sharing Git repositories remotely.

In addition to GitHub, there are other Git hosting services such as GitLab, Bitbucket

GitLab

By using Git and GitHub, you can access your code from any computer.



Josep Carreras
LEUKAEMIA
Research Institute

2. Basic Git workflow

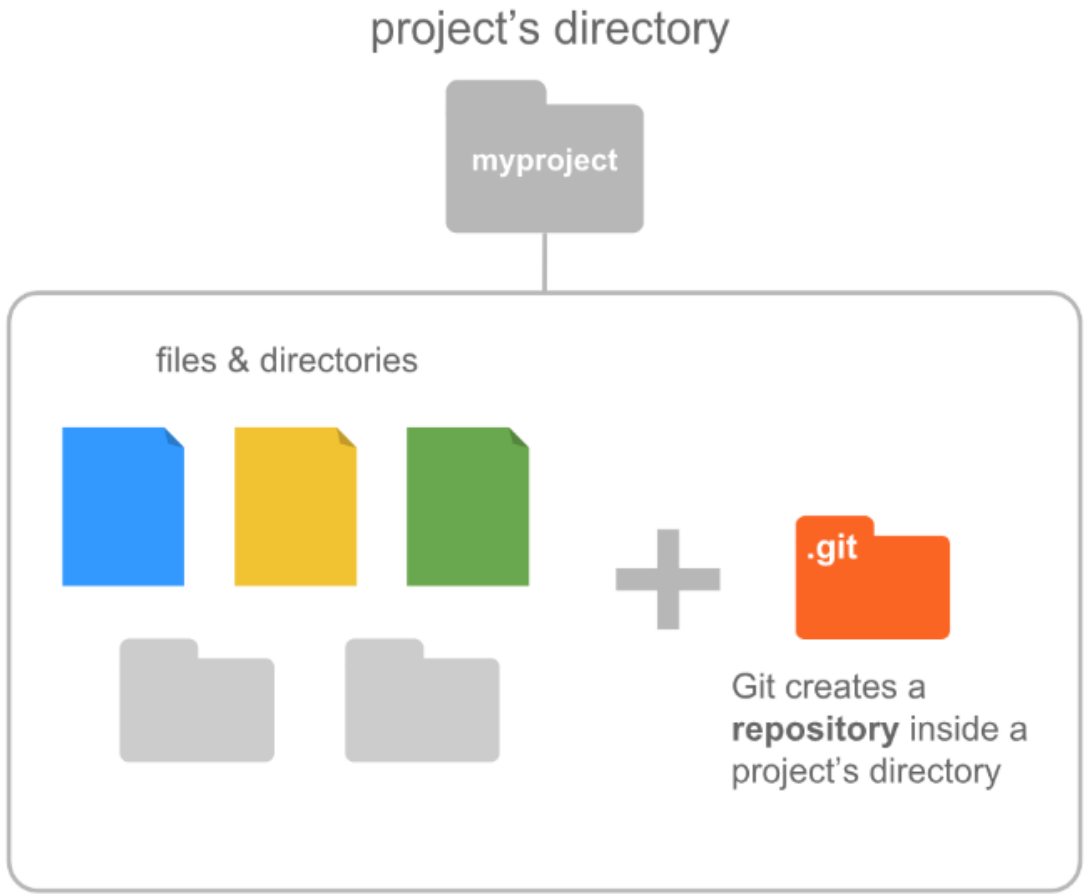
Basic Concepts

- A **repository** contains the history of your project. Find it in the **'.git'** subdirectory of your working directory.

- A **commit** is a snapshot of all the changes done and will be saved in the repository.

- The **staging area** stores information about what will go into your next commit.

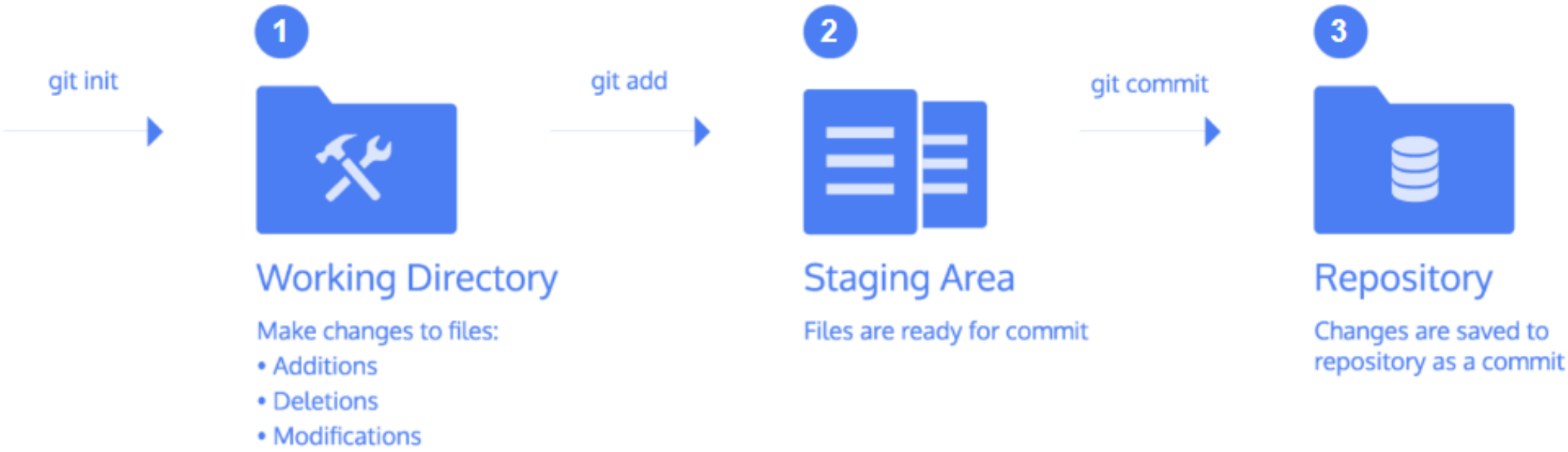
- A **branch** is an active line of development of your project.



Basic Git Workflow

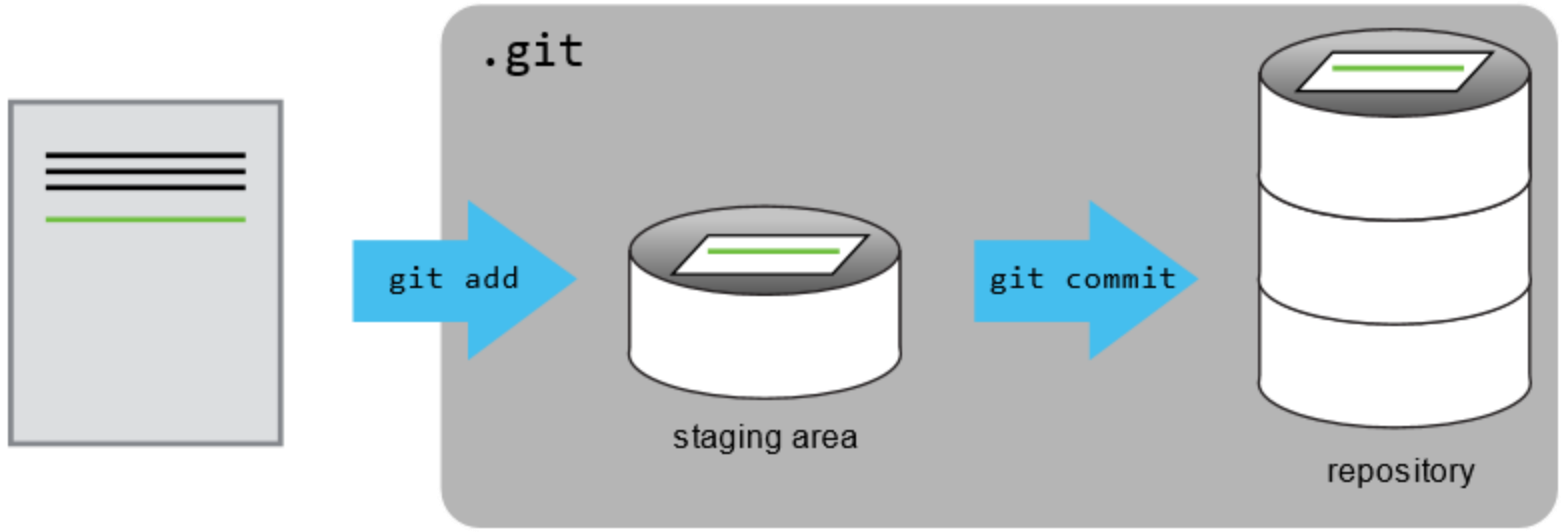
The basic Git workflow is the following:

1. **Modify** files in your working directory.
2. **Stage** just those changes you want to be part of your next commit, to the staging area.
3. **Commit**, which stores files into the Git repository.



Basic Git Workflow

Josep Carreras
LEUKAEMIA
Research Institute



- If a file was changed but not added to the staging area, it is **modified.**
- If it has been modified and added to the staging area, it is **staged**.
- If it is in the Git repository, it is **commited**.

Basic Git Workflow

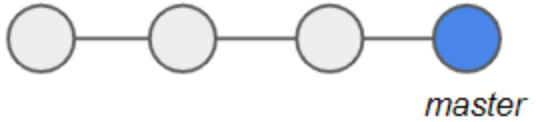
Josep Carreras LEUKAEMIA Research Institute

Git stores commits history

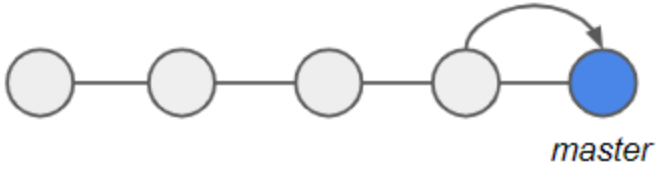
Used to **track changes, review, and revert to previous version**.

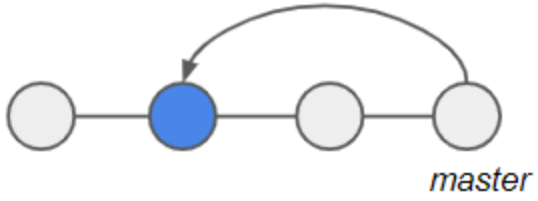
Basic Git Workflow

- You can consider your project history as a series of connected commits

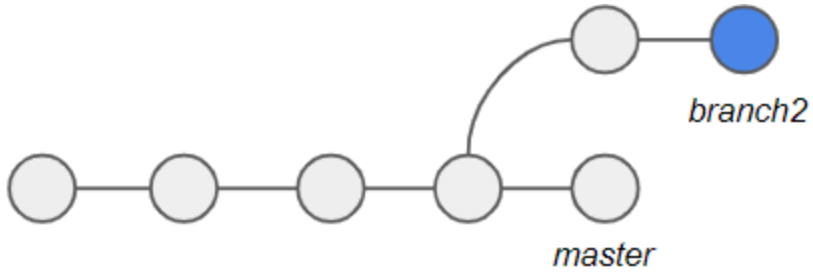


- You can keep adding new commits, check the previous ones, etc.





- However, this process is often **not linear** !



Josep Carreras
LEUKAEMIA
Research Institute

3. Basic Git Commands

Linux Terminal

Josep Carreras LEUKAEMIA Research Institute

Go to https://vpn.carrerasresearch.org/remote/login?lang=en

Enter your credentials and select the LINUX_COURSE bookmark.

Configure **Git username and email address:**

```
git config --global user.name "name"
git config --global user.email "email"
```

This information will be used to document who made changes to files in git. It is important to use the same email address and username that you setup on GitHub.

```
git config --list
```

Text editor – VIM

Josep Carreras LEUKAEMIA Research Institute

> ⚠ VIM
>
> If you haven't used Vim before, type **vim** "name of the file you want to create" on the command line.
>
> This will open you a new file. Press ESC and type i to change to the insertion mode. Finally press ESC and type :wq to exit the file.

Starting up a repository

Josep Carreras LEUKAEMIA Research Institute

Git init

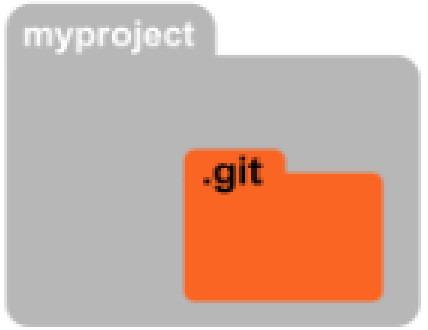
Initialize a git repository

```
git init
```

> Initialized empty Git repository in /your/path/

```
ls -a
```

> . git



Starting up a repository

Josep Carreras LEUKAEMIA Research Institute

Git status

Check the status of the project

```
git status
```

```
On branch master
No commits yet
Nothing to commit (create/copy files and use "git add" to track)
```

After creating or modifying a file:

```
git status
```

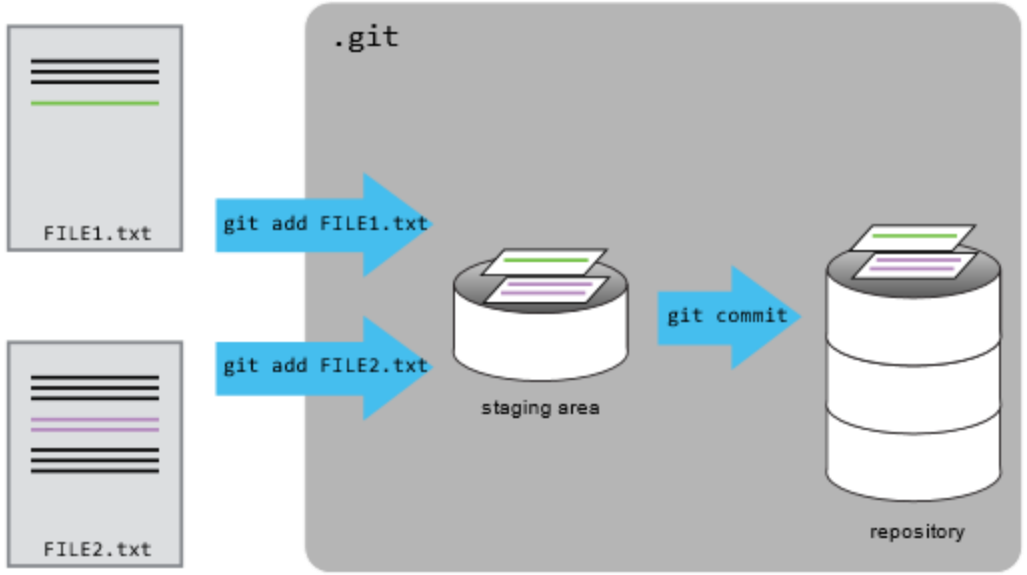
```
On branch master
No commits yet
Untracked files:
        sequence.txt
Nothing added to commit but untracked files present (use "git add" to track)
```

Git add

Add the file to the staging area

```
git  add  <file>
```

> **Tip**
>
> You can use git add <file1> <file2> to add more than one file to the staging are. To add all the files use git add *



Git commit

Stores changes from the staging area to the repository

```
git  commit –m "Short message of the changes"
```

```
[master (root-commit) f22b25e]  Descriptive message of the changes
1 file changed, 1 insertion (+)
Create mode 100644 sequence.txt
```

Exercises

Josep Carreras LEUKAEMIA Research Institute

1. Which command(s) below would save the changes of DNA_sequence.txt to my local Git repository?

1. `$ git commit –m 'my recent changes'`

2.
```
$ git  init   DNA_sequence.txt
$ git commit –m 'my recent changes'
```

3.
```
$ git  add   DNA_sequence.txt
$ git commit –m 'my recent changes'
```

4. `$ git commit –m  DNA_sequence.txt  'my recent changes'`

2.

-Create a new file named 'RNA_sequence.txt' and add some nucleotides.
-See the status of the project
-Commit the file RNA_sequence.txt and see the status of the project
-Modify both files by adding more nucleotides, and commit both files together.

3. What is the staging area used for?

Project History, review changes

Josep Carreras LEUKAEMIA Research Institute

Git log

Shows chronologically the project history of the **commits**

```
git log
```

```
commit f22b25e3233b4645dabd0d81e651fe074bd8e73b
Author: Marina <marina@marinavilardell.cat>
Start notes of a sequence
```

Commit identifier

Who made the changes

Commit message

Git diff

Shows the changes between the last commit and the working directory

```
git diff <file>
```

```
git diff commitID <file>
```

REMEMBER!

Josep Carreras LEUKAEMIA Research Institute

- Adding changes to the staging area:

  git add <file>

- Commiting the staged snapshot to the repository:

  git commit -m 'message'

- Files staged, unstaged and untraked:

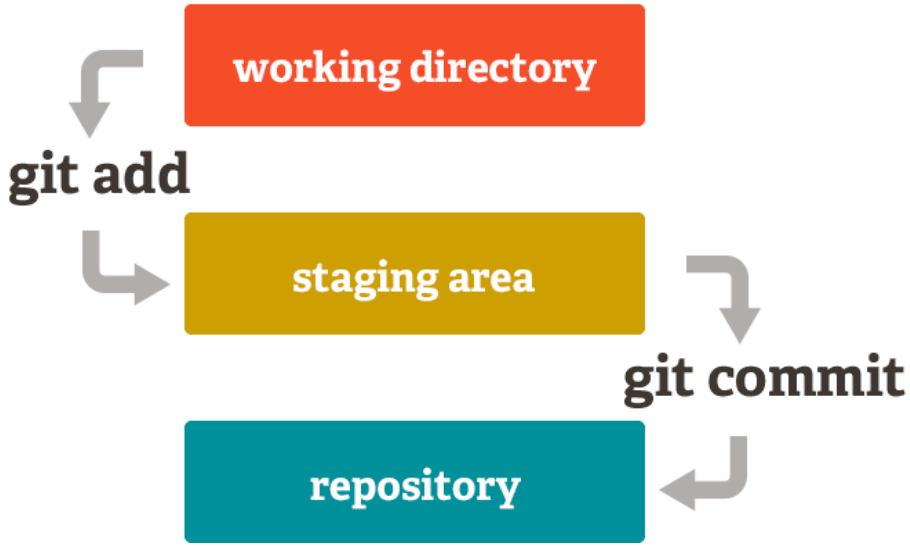
  git status

- Show the entire commit history:

  git log

- Show the changes of a file:

  git diff <file>



Exercise

Josep Carreras LEUKAEMIA Research Institute

4. Modify the first line of the RNA_sequence.txt file, and add another line containing the four nucleotides of a RNA sequence.
   Check the status.
   Display the differences of the RNA_sequence.txt
   Now, add the file into the staging area, and display again the differences. What happens? Why do you think this is happening?

   Finally, commit the changes and check the commit history.

5. Show the changes between the last commit and the first commit you did to the RNA_sequence.txt.

4. Backtracking

Undo changes

Josep Carreras LEUKAEMIA Research Institute

The HEAD commit

The most recently made commit is the HEAD commit.

```
git show HEAD
```

Git checkout

Restores the file in your working directory to look exactly as it did when you last made a commit

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

If you want to go back to an older version, substitute HEAD by the commit ID.

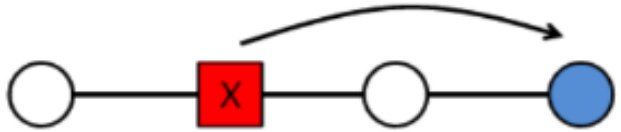
```
git checkout f22b25e <file>
git status
```

> **Note**
>
> git checkout returns the files not yet committed within the local repository

Git revert

Undoes a commit by creating a new commit

```
git  revert HEAD
```



Creates a new commit that reverts the changes

Other git commands:

- Removing untracked files from your working directory

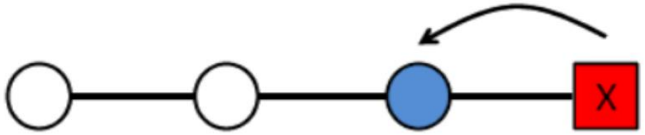
  `git clean`

- Fix the last commit (change commit message, add new files)

  `git commit --amend`

- Removing commited snapshot (dangerous way to undo changes)

  `git reset <commitID>`



Exercises

Josep Carreras LEUKAEMIA Research Institute

6. Write the following message in the DNA_sequence.txt file:

The four nucleotides of a DNA sequences are: A,C,T,G

Commit the changes.
Modify the file to have a U instead of a T, and commit the changes.
Now you realized the message was wrong.
Which commands can you use to undo the commit and restore the file to the previous version?

7. GETTING RID OF STAGED CHANGES:

Make a change to DNA_sequences.txt file, add that change to the staging area. Unstage the file and restore the file to the previous commit.

Exercises

Josep Carreras LEUKAEMIA Research Institute

8.

What is the output of the last command ?

```
$ echo "A gene is a sequence of DNA" > genes.txt
$ git add genes.txt
$ echo "A gene is a fundamental unit of heredity in living organisms" >> genes.txt
$ git commit -m "Simple Gene description"
$ git checkout HEAD genes.txt
$ cat genes.txt
```

1. A gene is a fundamental unit of heredity in living organisms

2 . A gene is a sequence of DNA

3. A gene is a sequence of DNA.
   A gene is a fundamental unit of heredity in living organisms.

4. Error because you have changed genes.txt without commiting the changes.

Josep Carreras
LEUKAEMIA
Research Institute

5. Remote repositories

Remote repositories

Josep Carreras LEUKAEMIA Research Institute

Remember

Remote repositories are hosted on a server (internet).
All the modifications we have done until now are committed locally. Then, those changes have to be uploaded to a remote repository.

Go to GitHub and sing up.
Then, create an empty repository.

Create a connection to the remote repository:

```
git remote add <name> <SSH link>
```

To list connections:

```
git remote -v
```

Other commands:

`git remote rm <name>` remove connection

`git remote rename <old> <new>` rename connection

SSH configuration

Josep Carreras LEUKAEMIA Research Institute

Create a SSH Key:

```
ssh-keygen -t ed25519 -C  user.email
```

Copy the key to GitHub :

```
cat ~/.ssh/id_ed25519.pub
```

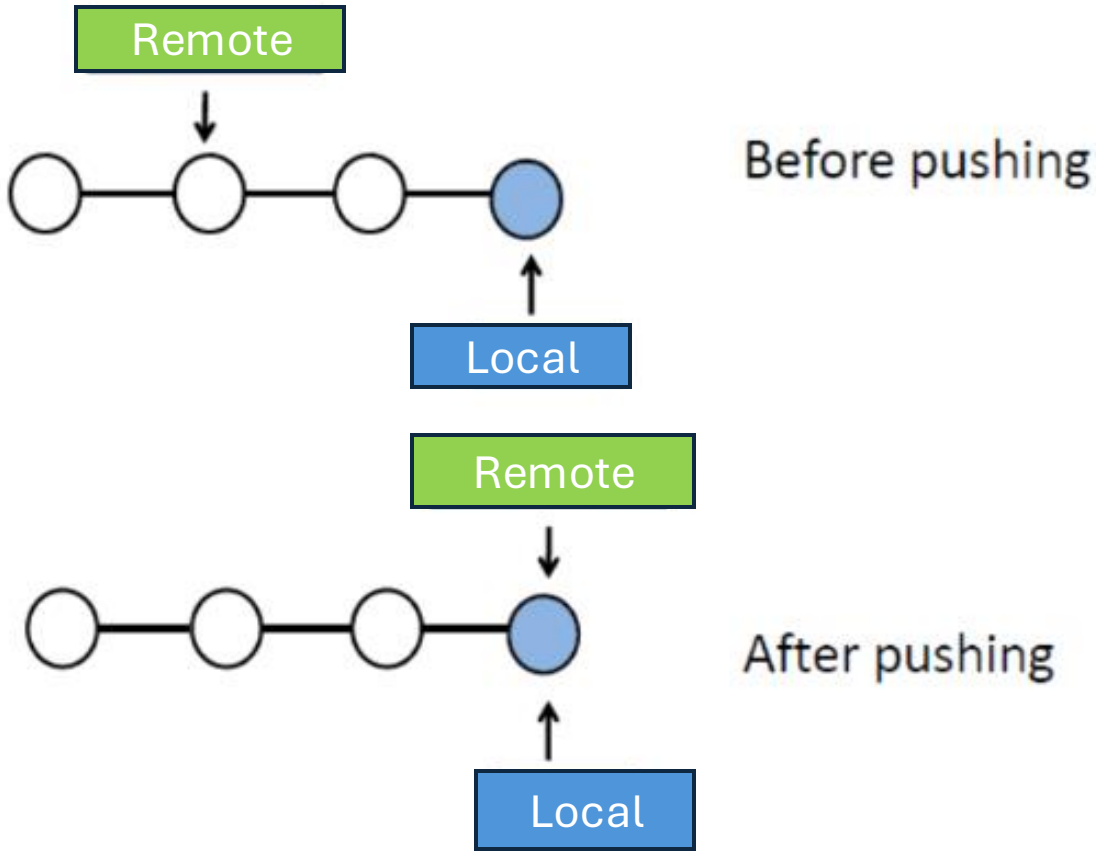
Check the authentication:

```
ssh -T git@github.com
```

Git push

Transfer commits from a local repository to the remote repository (create a copy of the local in the remote repository)

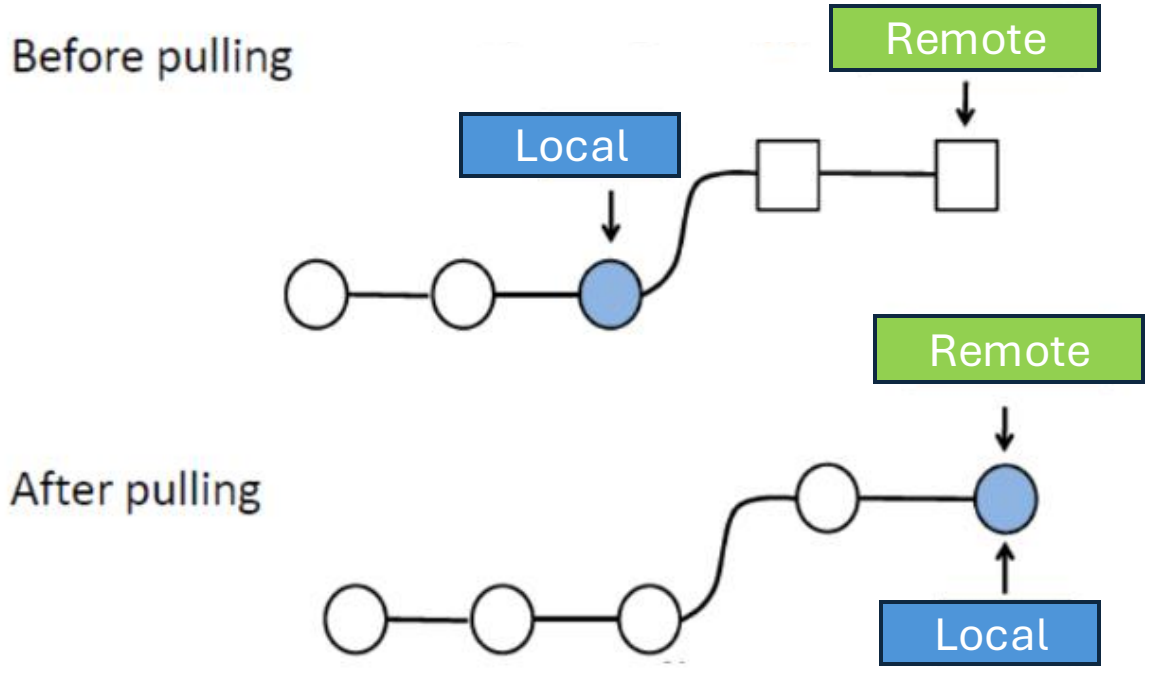
```
git push  <remote>  master
```

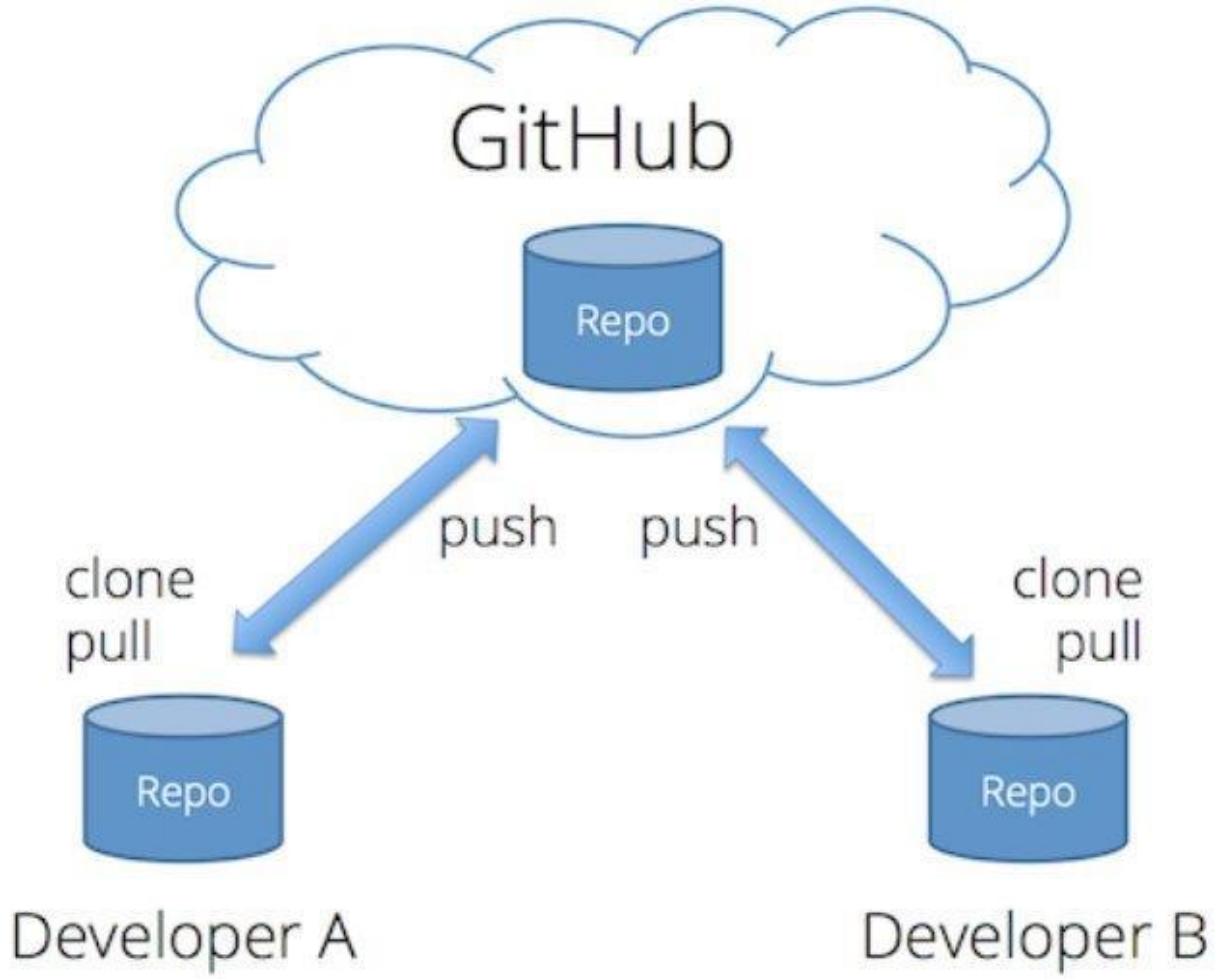


Git pull

Import commits from the remote repository to a local repository

```
git pull <remote> master
```





Git clone

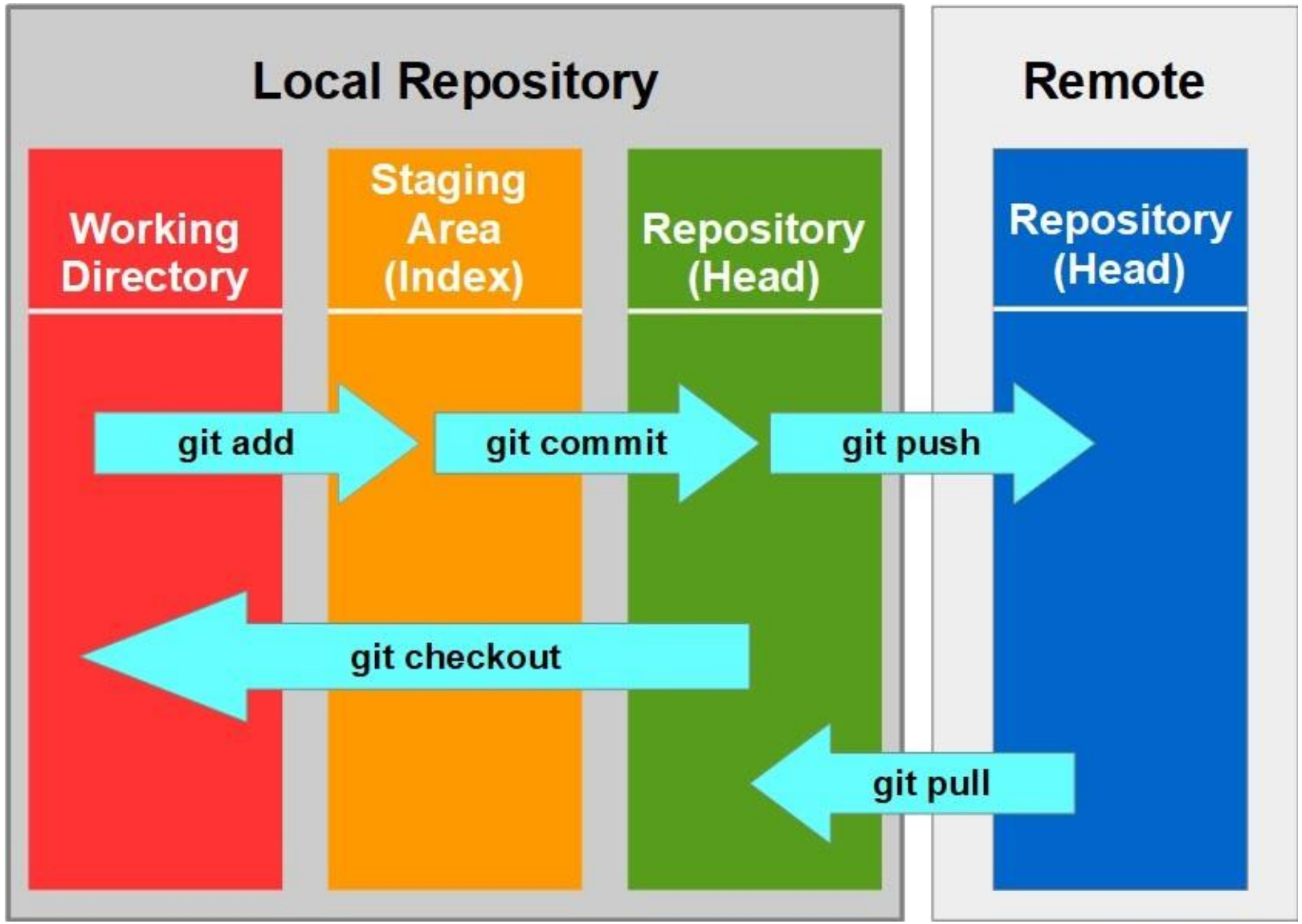
```
git clone <URL>
```

Differentiate git clone and git pull:

- git clone creates a copy of all the files of the remote repository to the local.
- git pull Updates the local repository by copying only the modified files of the remote repository.

Basic commands summary

Josep Carreras
LEUKAEMIA
Research Institute



Josep Carreras
LEUKAEMIA
Research Institute

6. Mantaining a good repository

Best practices to maintain a good repository

Josep Carreras LEUKAEMIA Research Institute

This tips will help you maintain a clean, efficient, and well-managed repository:

1. **Use a clear repository name**
2. **Create a README.md file**
3. **Use branches in a consistent strategy**
4. **Maintain a clean commit History**
5. **Utilize a .gitignore file**

Resources to learn more

Josep Carreras LEUKAEMIA Research Institute

Official Git site

https://git-scm.com/doc

This workshop is based on:

https://osulp.github.io/git-beginner/

https://osulp.github.io/git-advanced/

Other references

https://www.atlassian.com/git/tutorials

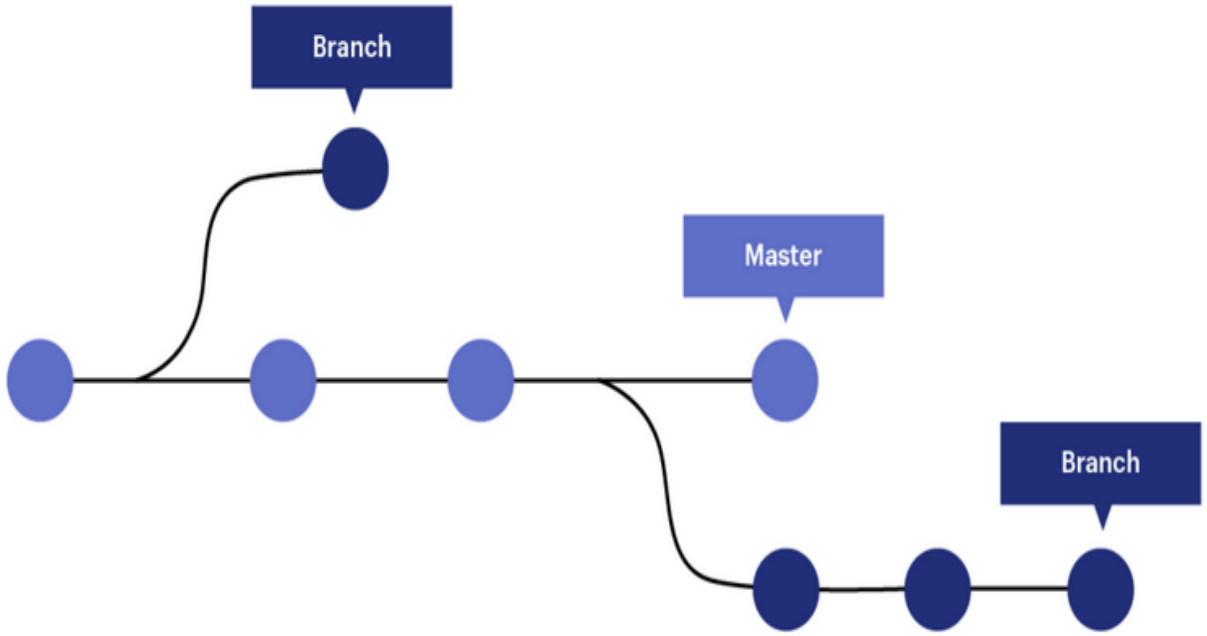
https://docs.github.com/en

7. Branches

Branches

Josep Carreras LEUKAEMIA Research Institute

A branch represents an independent line of development (a new working directory, staging area and project history). In other words, a new/separate version of the main repository.



Why should you use branches?

Branches allow you to develop features, fix bugs, or safely experiment with new ideas in a contained area of your repository.

Use a branch to isolate development work without affecting other branches of your repository.

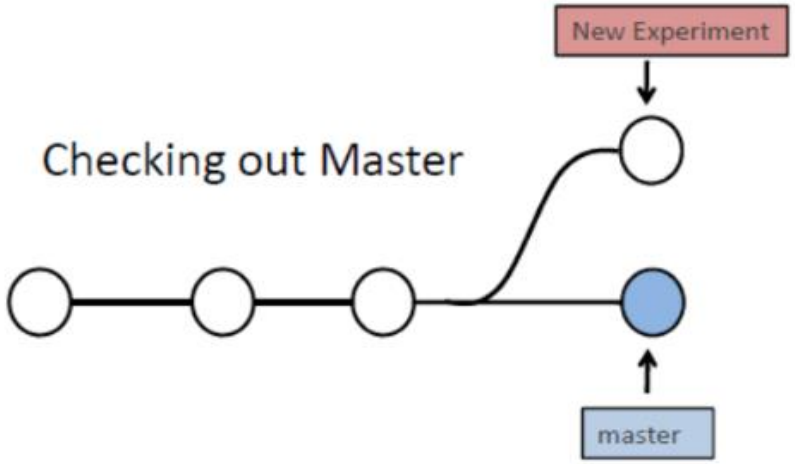
Branches basic commands

Josep Carreras
LEUKAEMIA
Research Institute

`git branch`	list all branches
`git branch <branch>`	create new branch
`git branch -d <branch>`	delete

Navigating between branches:

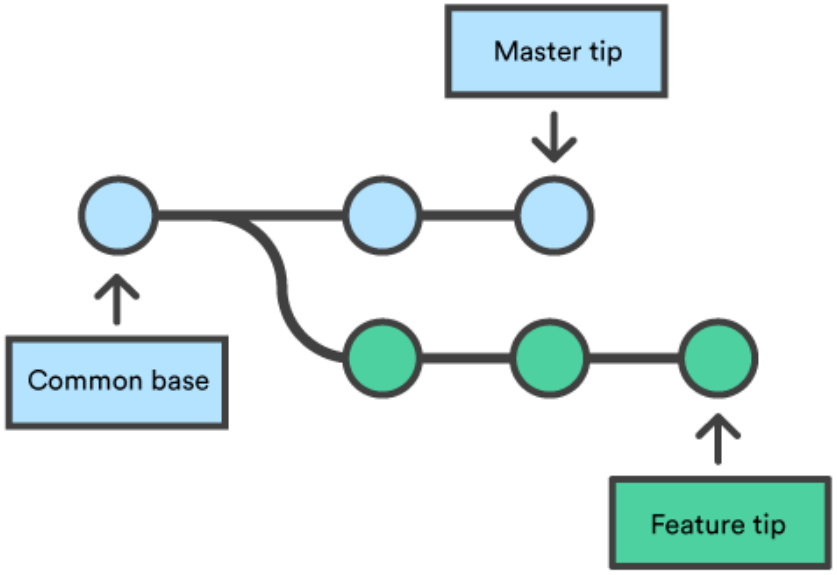
`git checkout <branch>`



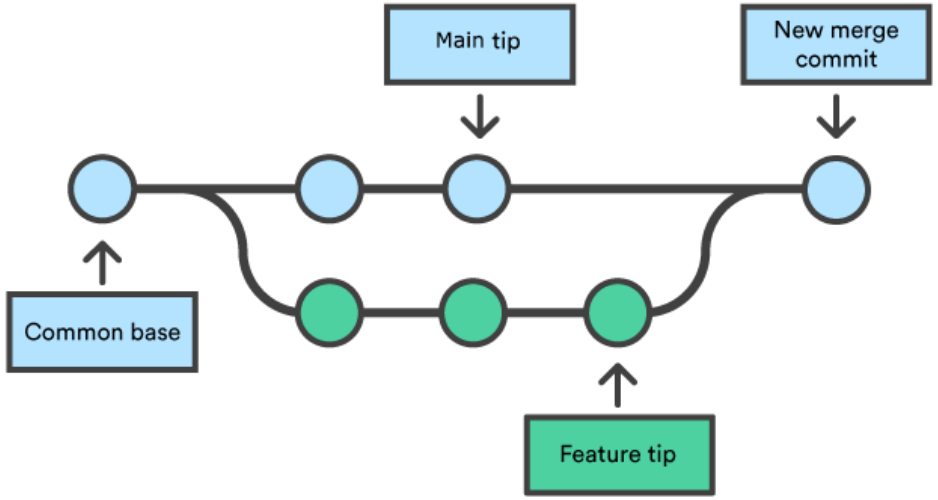
Merging

Josep Carreras LEUKAEMIA Research Institute

Before merging branches



After merging branches



When the work is complete, a **branch can be merged** with any other branch in your Git repository.

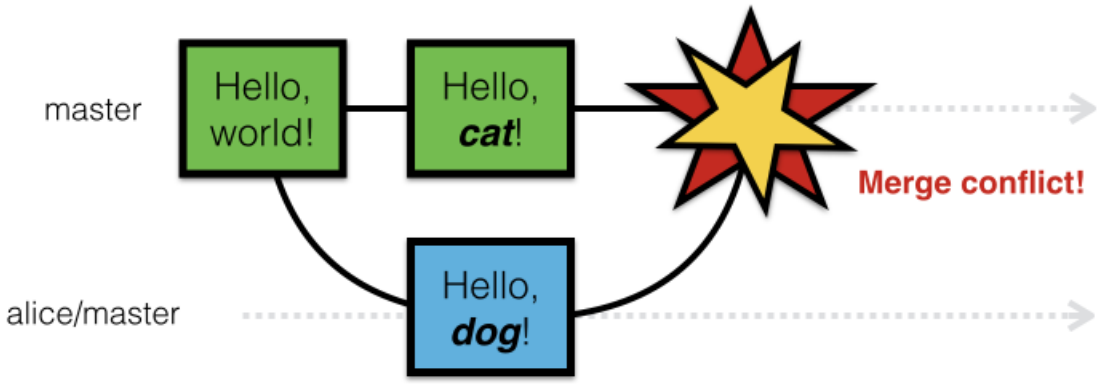
```
git merge <branch>
```

You can even switch between branches and work on different projects without them interfering with each other.

Merge conflicts

Josep Carreras LEUKAEMIA Research Institute

If two branches change the same part of the same file, Git stops right before the merge commit.



CONFLICT !!

Git needs your help to decide which changes to incorporate in the final merge

Conflicts must be resolved manually!

`git status` Shows which files need to be resolved