



**GIT  
WORKSHOP**

**JUL 07 - 2020**

# **BACK TO THE FUTURE**

# PREREQUIRES



YOU HAVE GIT ON YOU  
MACHINE



YOU HAVE GITLENS  
VSCODE EXTENSION



YOU HAVE A GIT  
ACCOUNT



WORK ON **MY OWN** COMPUTER WITH GIT

TO **COMPARE** TO HISTORY

INIT

CLONE

ADD

COMMIT

PUSH

DIFF

CHECKOUT

TO **REVERT** CHANGES

RESET

WORK WITH **OTHER PEOPLE** WITH GITHUB

IN 1 **BRANCH** WITH NO CONFLICT

PULL

IN 1 BRANCH WITH **CONFLICTS**

IN **MANY BRANCHES**

BRANCH

MERGE

**RELEASE** APPLICATIONS

TAG

**JUL 07 - 2020**

1

**YOU LISTEN AND UNDERSTAND**

2

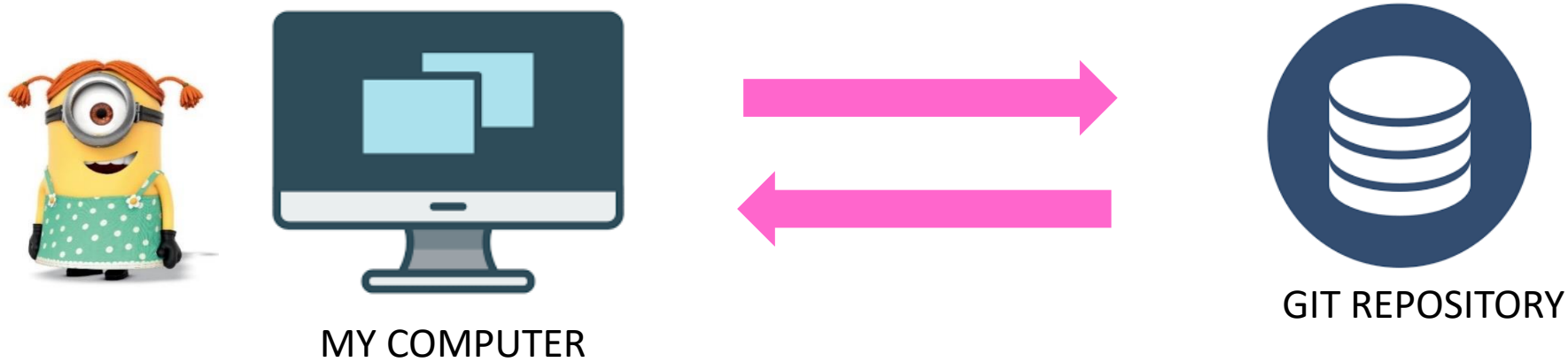
**YOU DO IT**

#1 to #19

INDIVIDUAL WORK



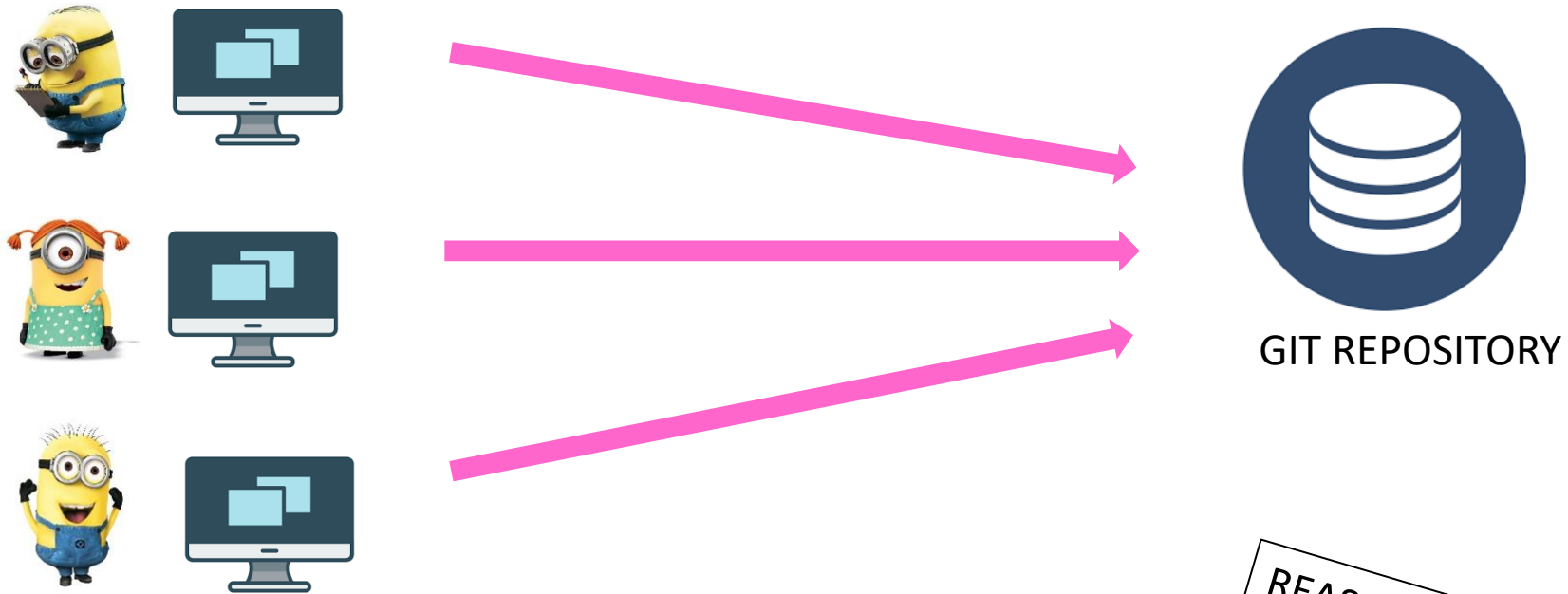
# WHAT GIT IS USED FOR ?



REASON #1

BE ABLE TO SEE MY **HISTORY** OF CHANGES  
AND MAYBE **UNDO** SOME OF THEM

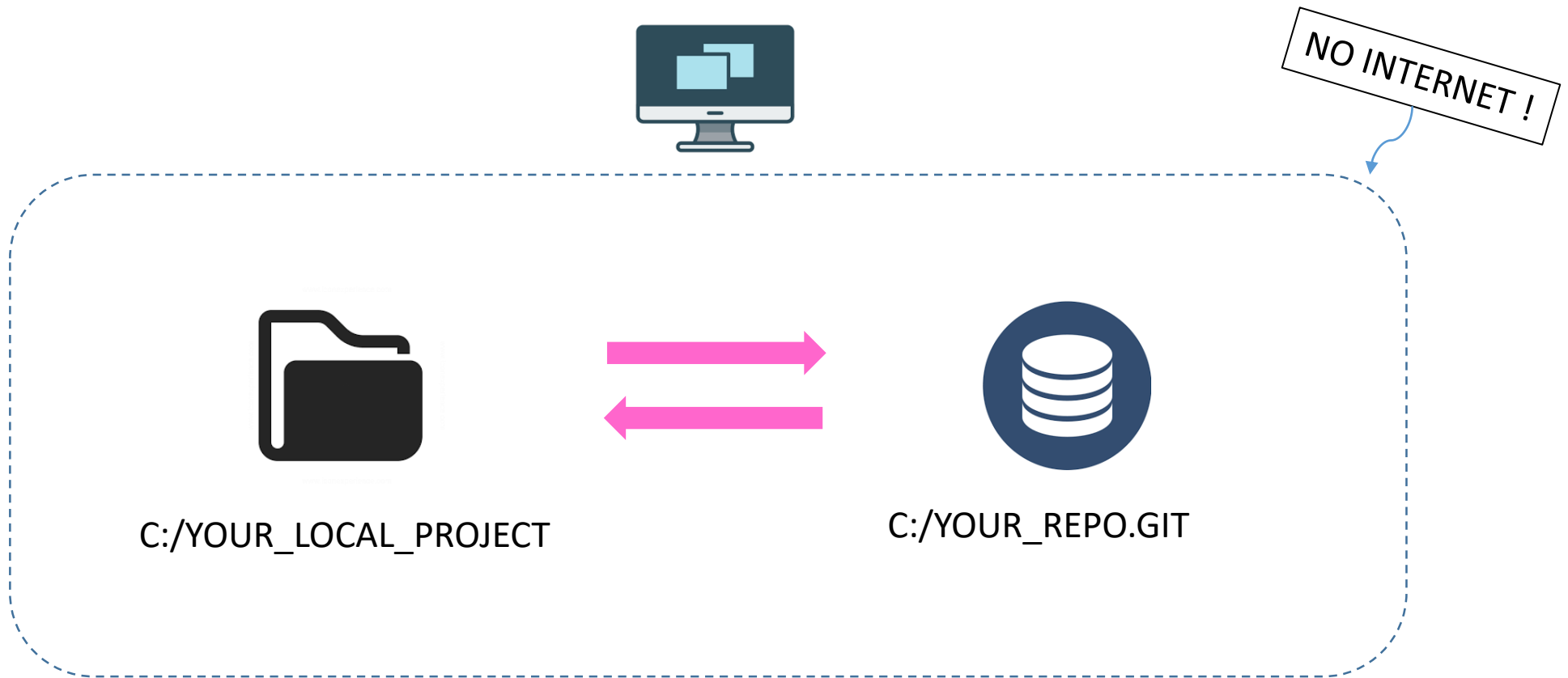
# WHAT GIT IS USED FOR ?



REASON #2

BE ABLE TO **WORK TOGETHER**  
ON A SAME PROJECT

# WHAT IF IS GIT WHAT JUST ON YOUR COMPUTER !!!!





# #1 Initialize a GIT repository

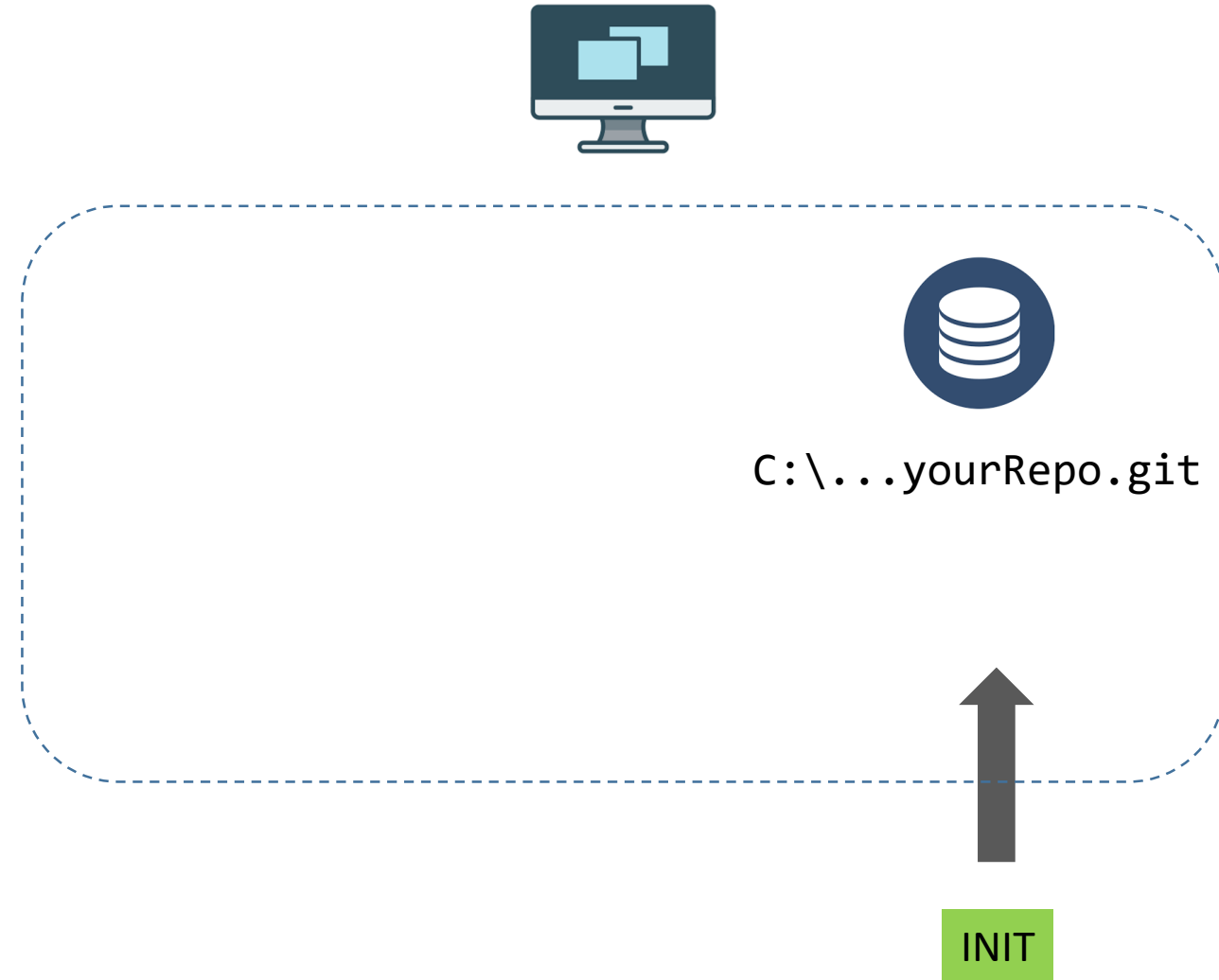
1 – Create a folder to store the repo GIT

C:\myGitServer\

2 – Run init command from this folder:

```
git init yourGitRepo.git --bare
```

✓ Check you GIT repository has been created



## #2 Clone this repo to a folder

1 – Create a folder to start working

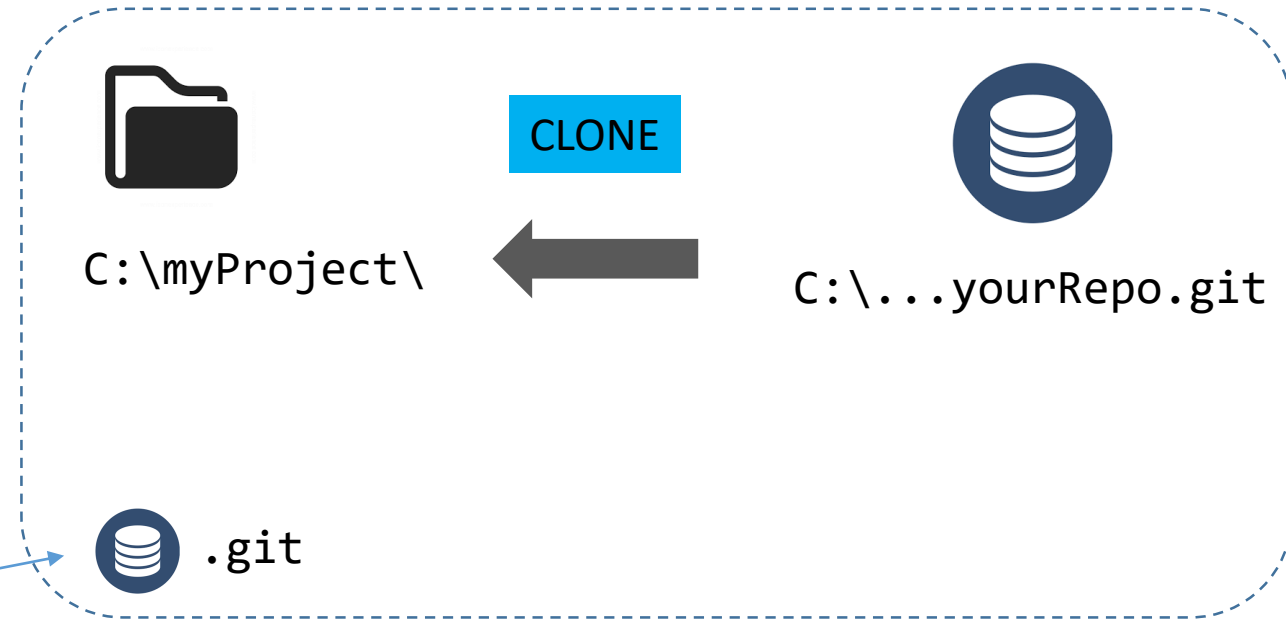
C:\myProject

2 – Clone the repository

```
git clone c:/..yourGitRepo.git
```

⚠ Direct slashes

✓ Check an folder is created with a .git folder



## #3 Just add a new file

1- Open this folder on VS Code

2- Check status

`git status`

✓ Check status : nothing changed

3- Create a new file: `index.html`

```
<!DOCTYPE html>
<html lang="en">
  <body>
    <h1>My first commit</h1>
  </body>
</html>
```

4- Check status

`git status`

✓ Check status : 1 new file to add



C:\myProject\

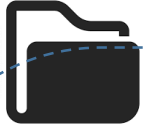
+ index.html



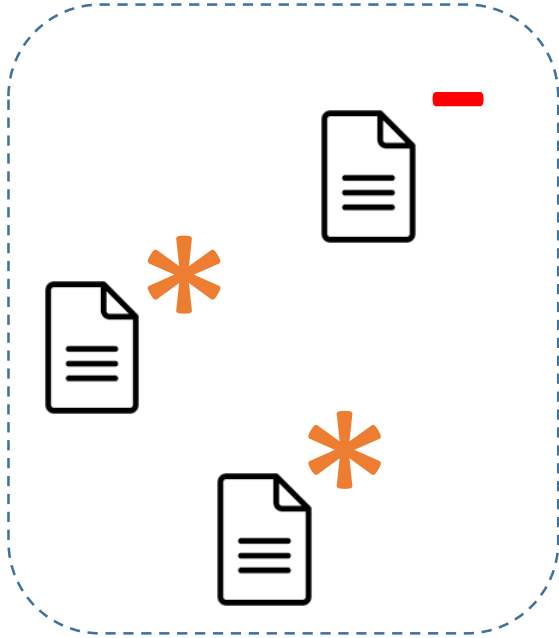
.git



C:\...\yourRepo.git



C:\myProject\



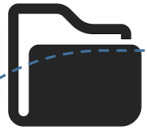
< staging area >



.git



**WE WANT TO COMMIT ONLY THOSE FILES**



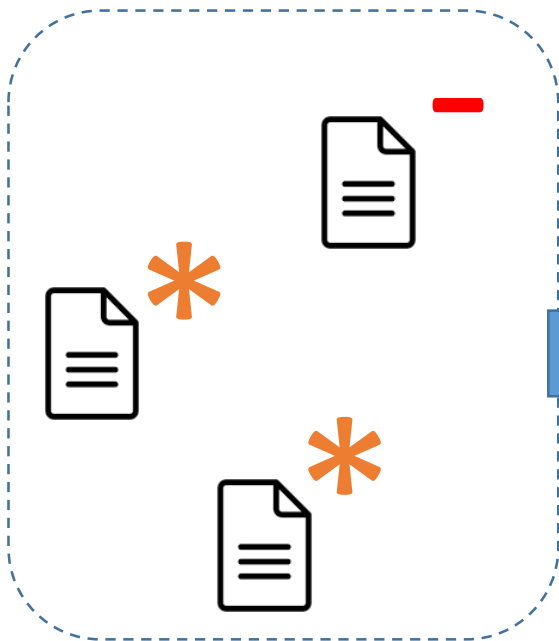
C:\myProject\

TEMPORARY AREA  
BEFORE COMIT

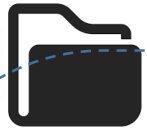
< staging area >



.git



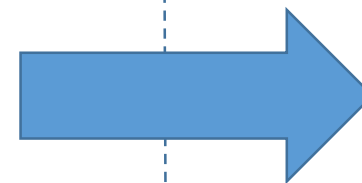
**WE ADD THOSE FILES TO THE STAGING AREA**



C:\myProject\

TEMPORARY AREA  
BEFORE COMIT

< staging area >



**WE COMMIT, WITH A COMMENT...**

## #4 Add this file to the stage and commit

1- Add file to stage

```
git add *
```



Check status : 1 change to be committed

2- Commit change

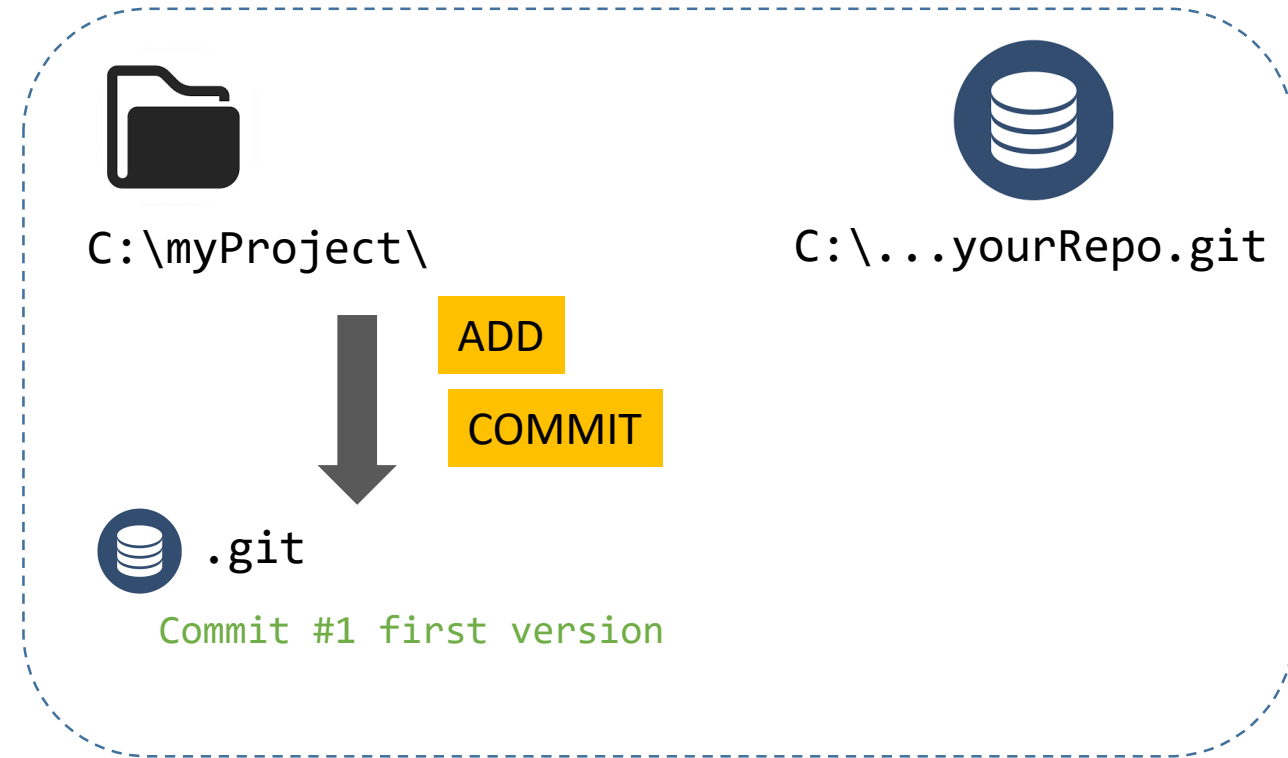
```
git commit -m "first version"
```



Check status : nothing to change



Check what is the id you your commit?





**DO IT !**

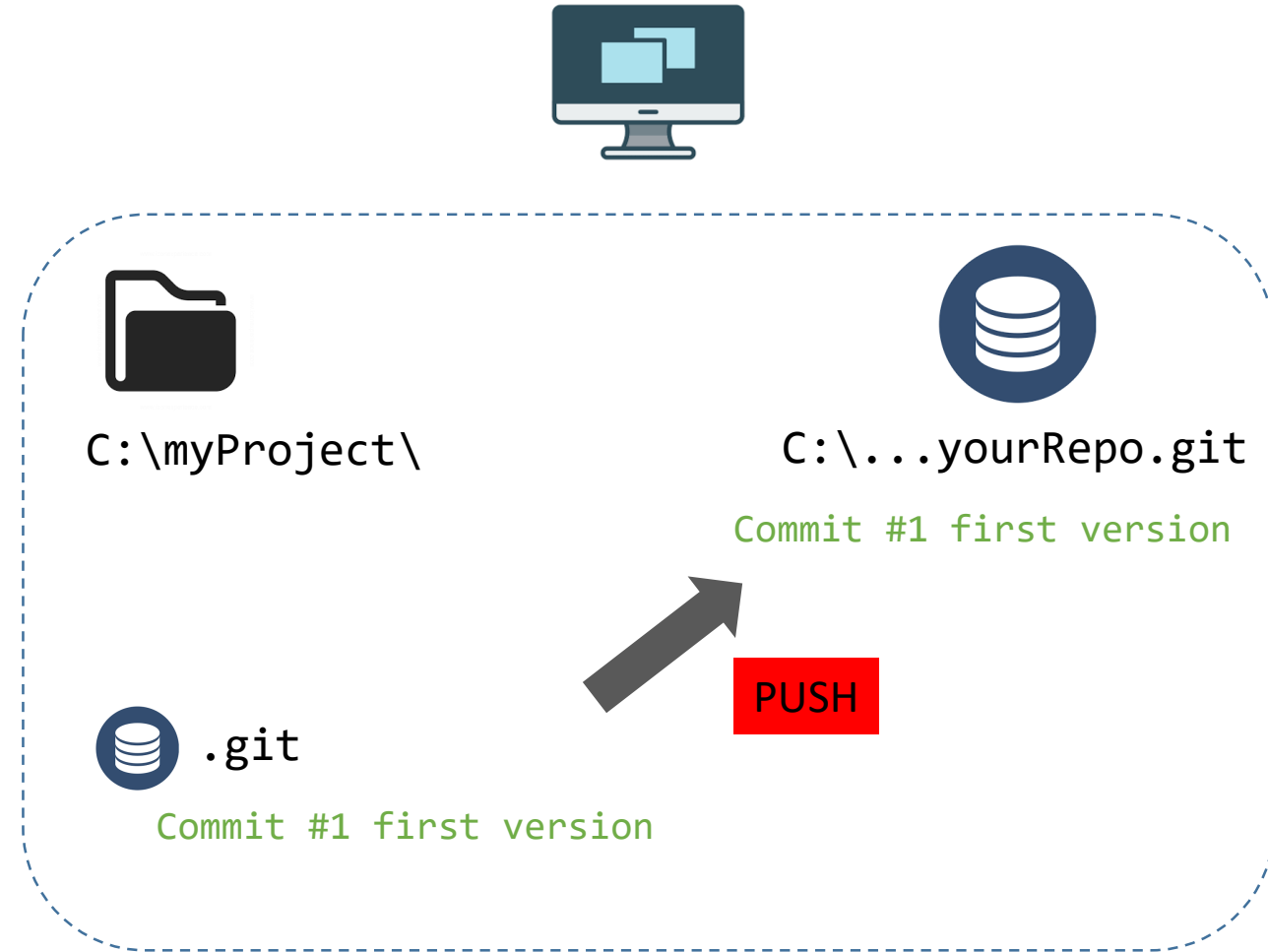


## #5 Push to your remote repo...

1- Push

`git push`

✓ Check the remote origin in VS Code Contains your commit



## #6 Let's add 2 changes...

### 1- Create style.css

```
.title {  
  background-color: blue;  
}
```

### 2- Update index.html

```
<!DOCTYPE html>  
<html lang="en">  
  <head>  
    <link rel="stylesheet" href="style.css" />  
  </head>  
  <body>  
    <h1 class="title">My app is yellow !</h1>  
  </body>  
</html>
```

### 3- Check status

✓ Check status : 2 changes



C:\myProject\

+ style.css  
\* index.html



C:\...\yourRepo.git

Commit #1 first version



.git

Commit #1 first version

## #7 Let's commit 2 changes...

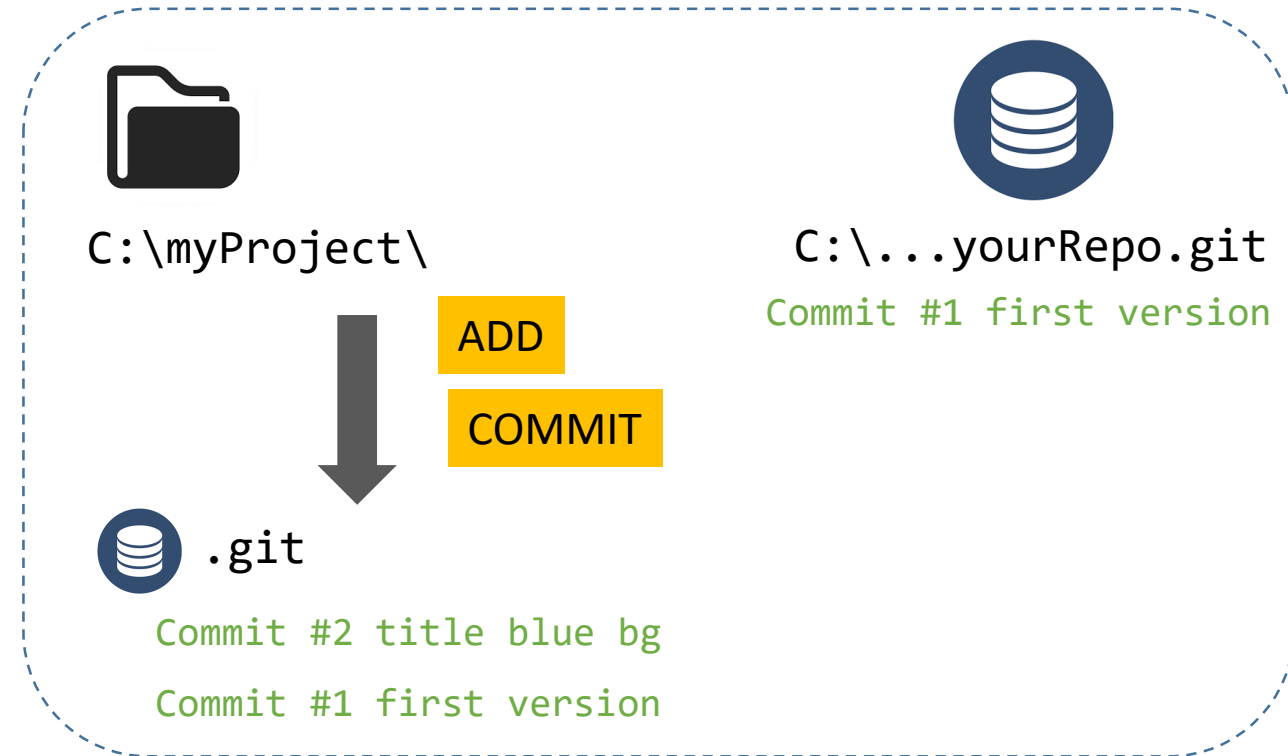
### 1- Add and commit

```
git add *
```

```
git commit -m "title blue bg"
```

✓ Check status : nothing to change

✓ Check what is the id you your commit?

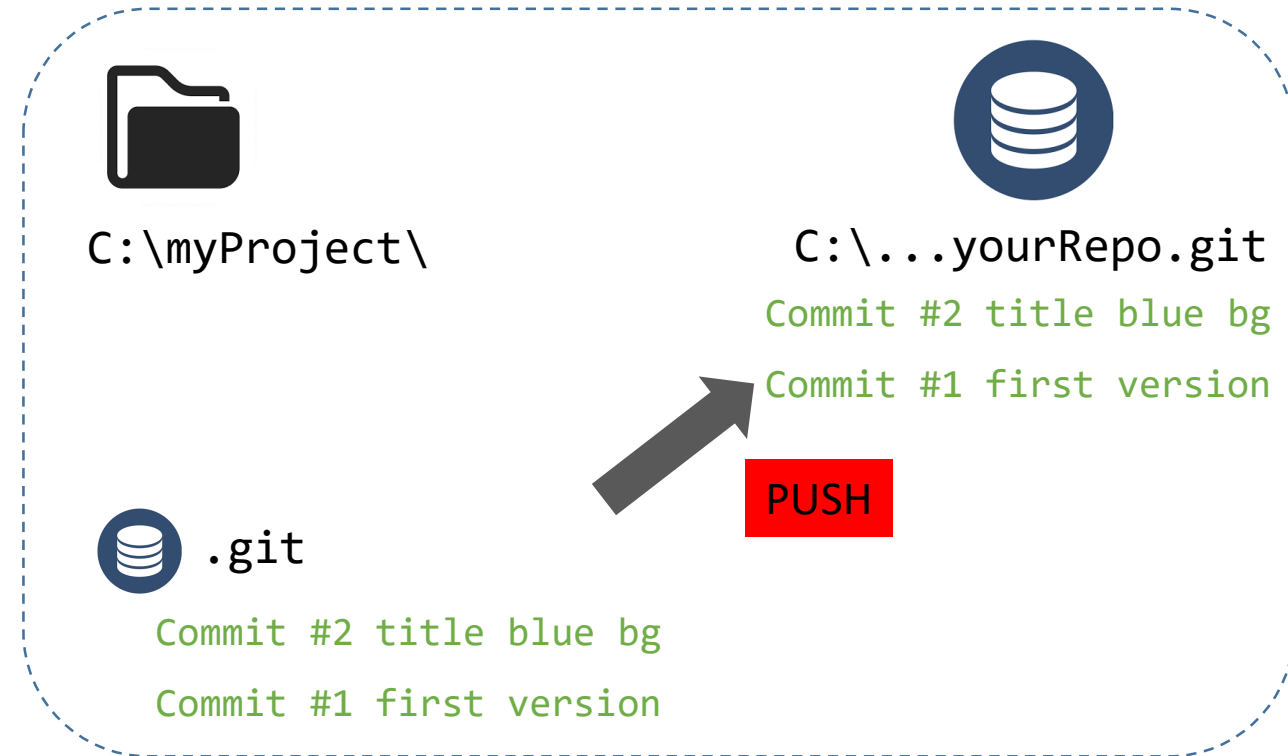


## #8 Let's push 2 changes...

1- Push

git push

✓ Check the remote origin in VS Code Contains your commit



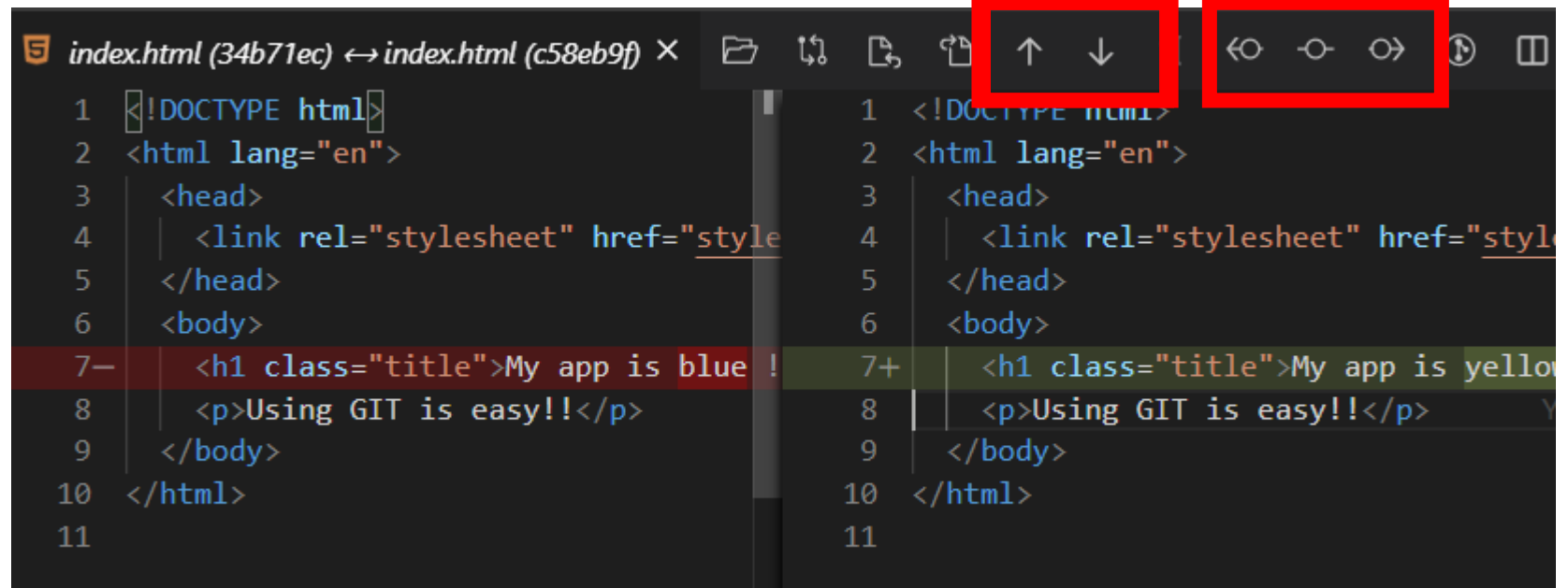
## #9 Use VS Code (+GitLens) efficiently

### 1- Commit + Push 3 times a change on the <H1>

1. My app is blue
2. My app is green
3. My app is yellow

### 2- Open the index.html in VSCode

✓ Check you can set all commit changes on this file by using the history arrows





**DO IT !**

## #10 Let's remove and restore

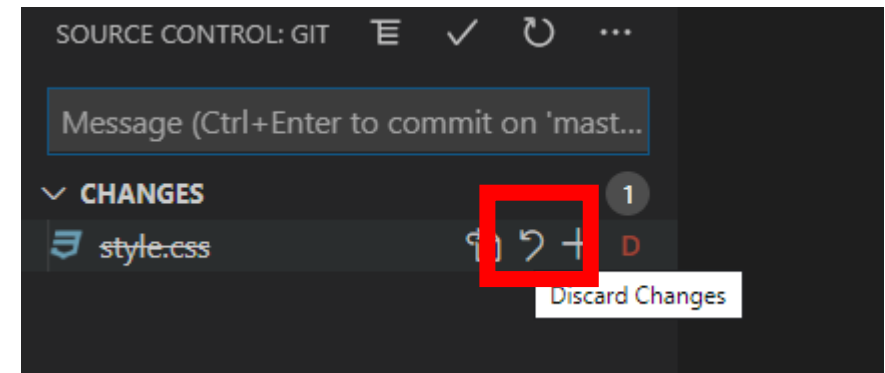
1- Remove style.css

2- Restore this file using command line  
`git reset --hard`

✓ Check file is restored

3- Remove again and restore this file using VS CODE

✓ Check file is restored



# #11 Let's update and restore

## 1- Update style.css

```
.title {  
  qdsdsqdsqdsq  
}
```

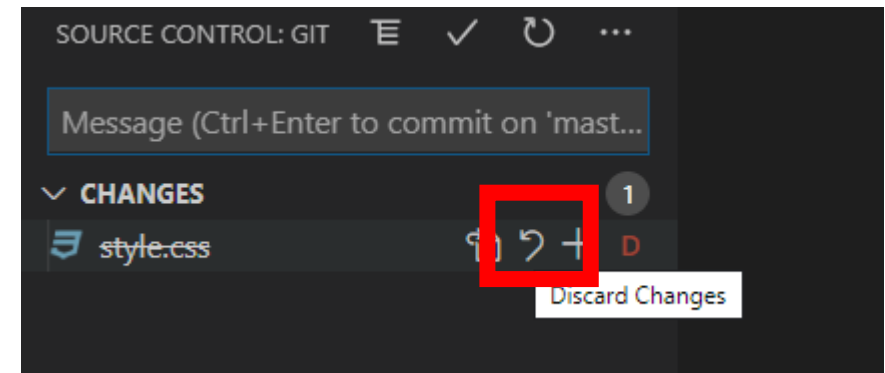
## 2- Restore this file using command line

```
git reset --hard
```

✓ Check file is restored

## 3- Remove again and restore this file using VS CODE

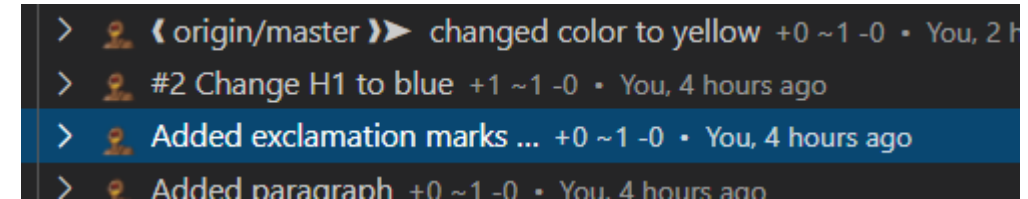
✓ Check file is restored



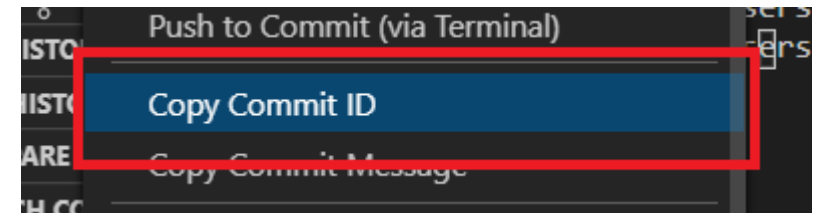


## #12 Let's compare 2 commits

1- Select a commit in GitLens view



2- Copy the commit ID



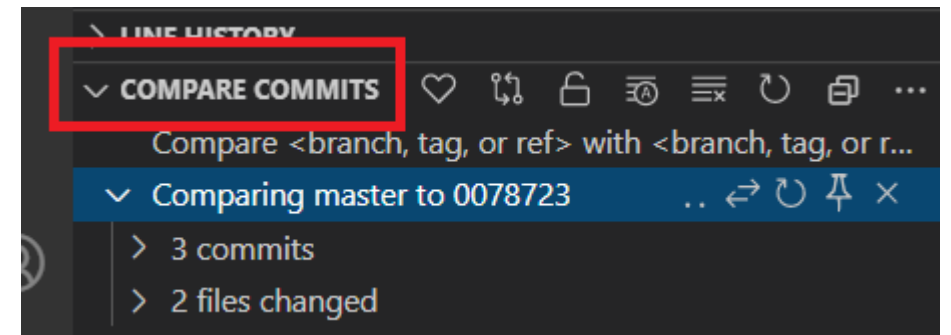
3- Once you have 2 commit, compare them:

```
git diff <id1> <id2>
```

⚠ To quit the diff mode : type 'q'

✅ Check you have see the differences between the 2 commits

4- You can also use VSCode to compare !



## #13 Let's clone again – on your local machine



1 – Create another folder

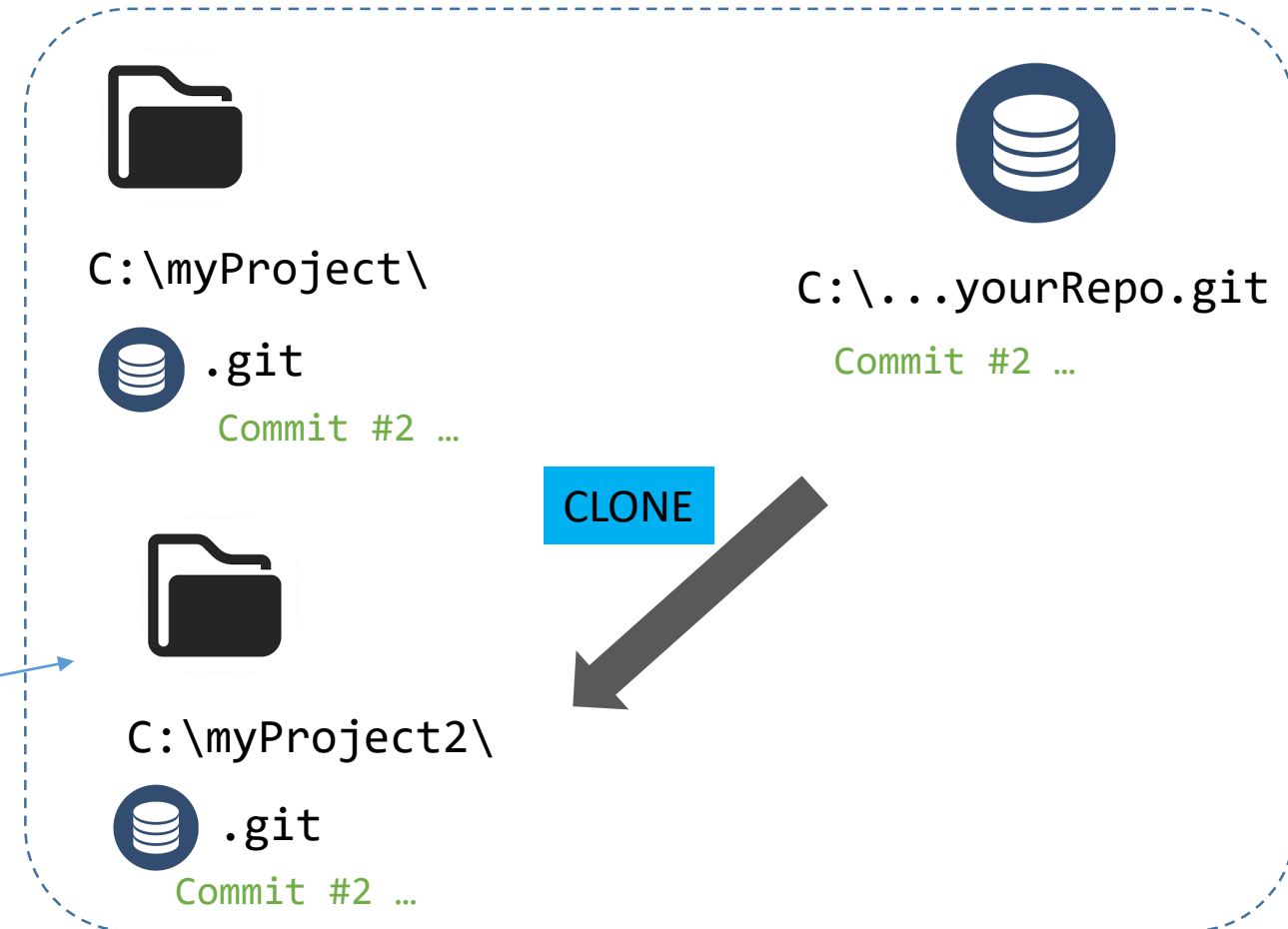
C:\myProject2

2 – Clone the repository

```
git clone c:/../yourGitRepo.git
```

⚠ **Direct slashes**

✓ Check an folder is created with a .git folder



## #14 Push a change from project 1...

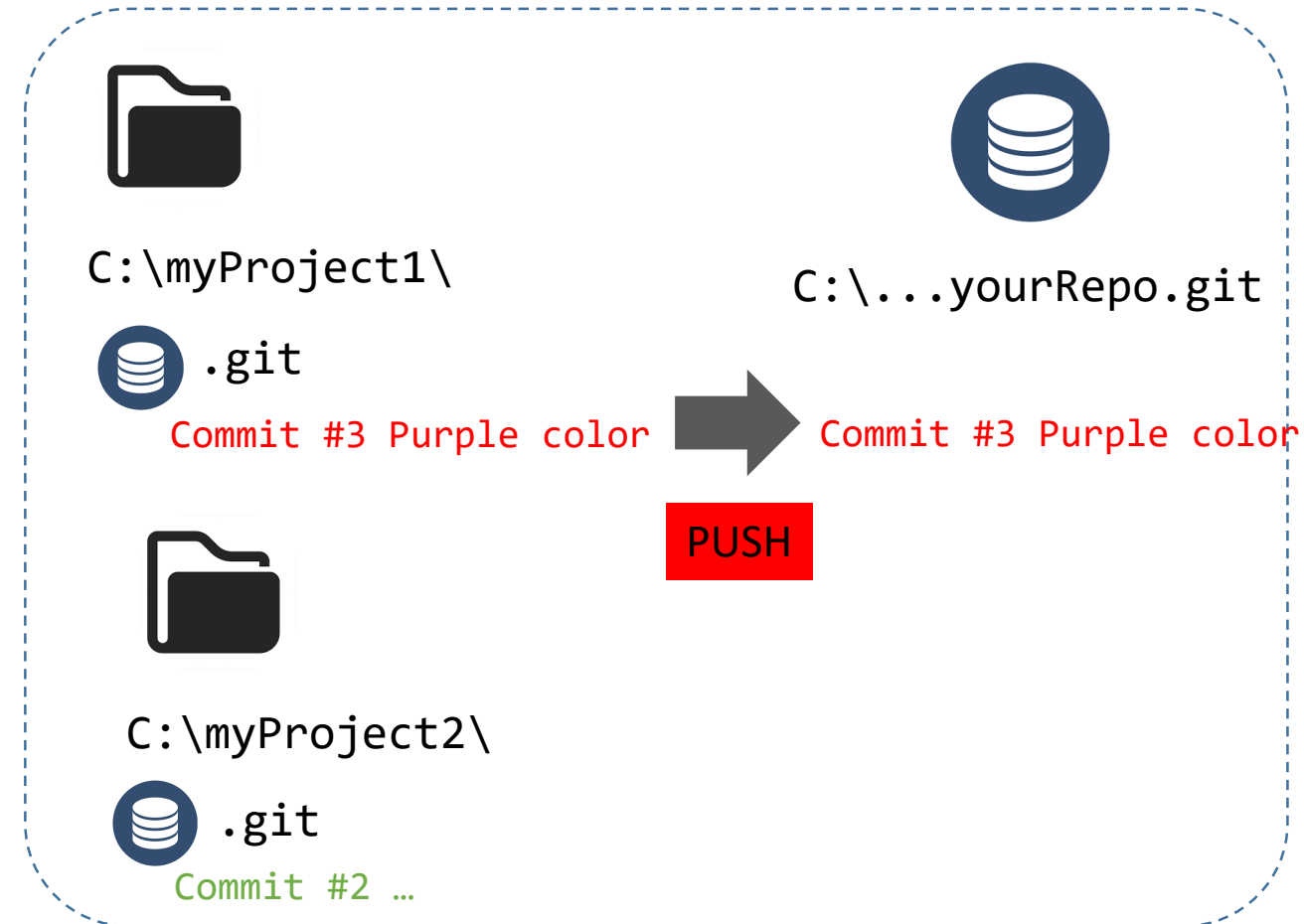
1 – From project1, edit a change on style.css

```
.title {  
  background-color: purple;  
}
```

2 – commit and push : “purple color”

✓ Check remote origin has the new commit

✓ Check on project2, the new commit is still not present



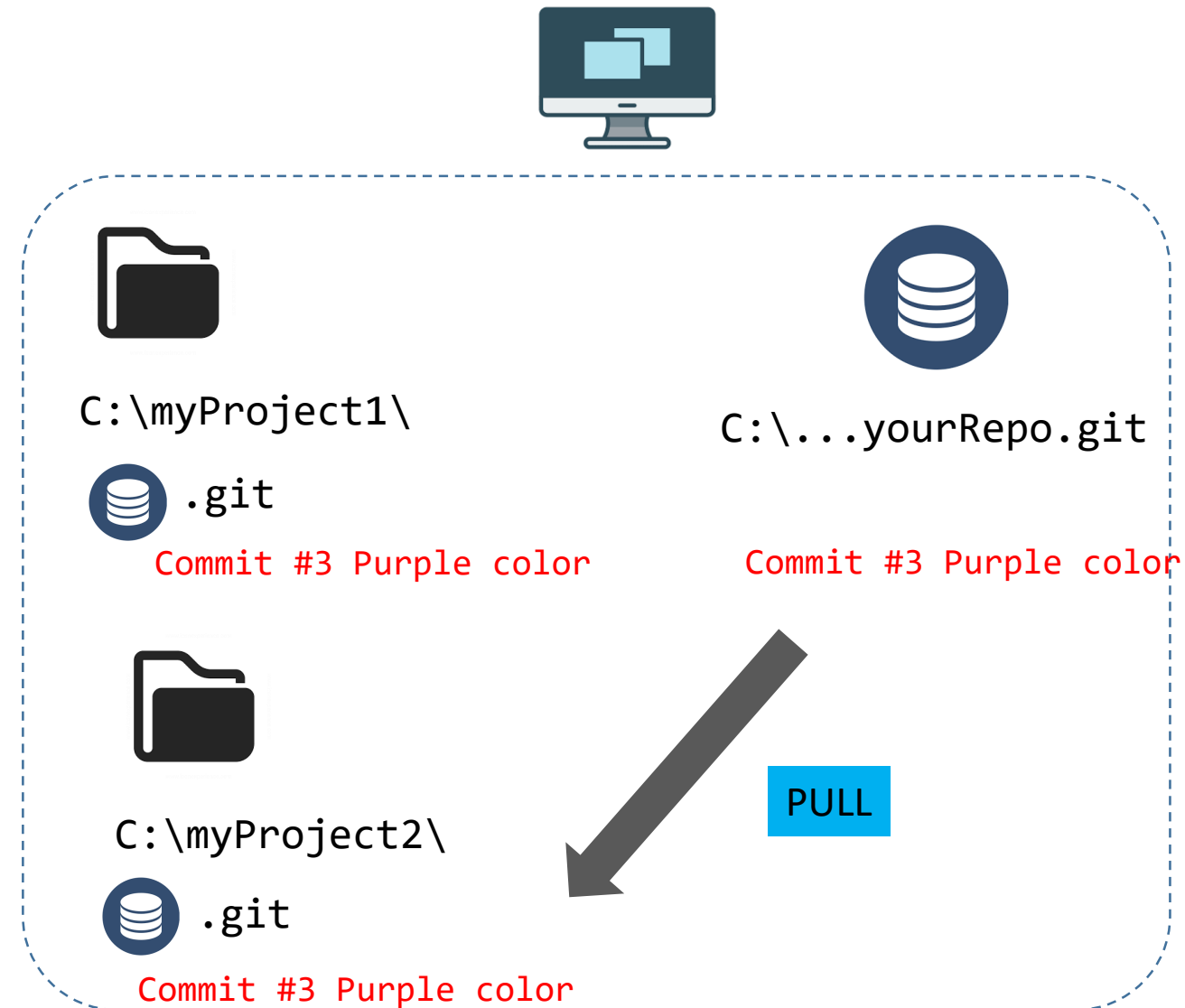
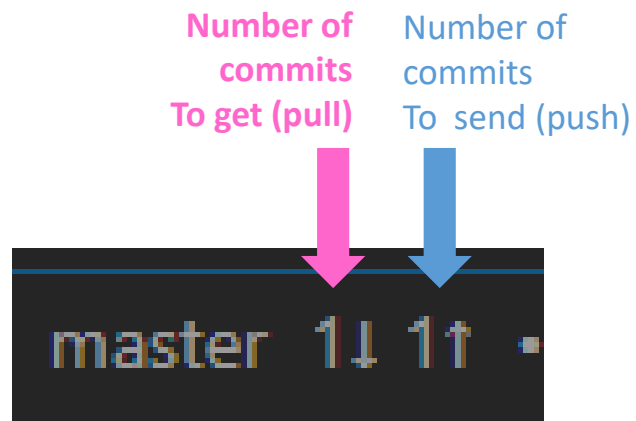
## #15 Fetch and pull on project2

1 – From project2, fetch

✓ You should see you have 1 commit behind

2 – From project2, pull

✓ Now on project2, the new commit has been added





**DO IT !**

## #16 Let 's create a conflict

1- From **project1** , commit + push

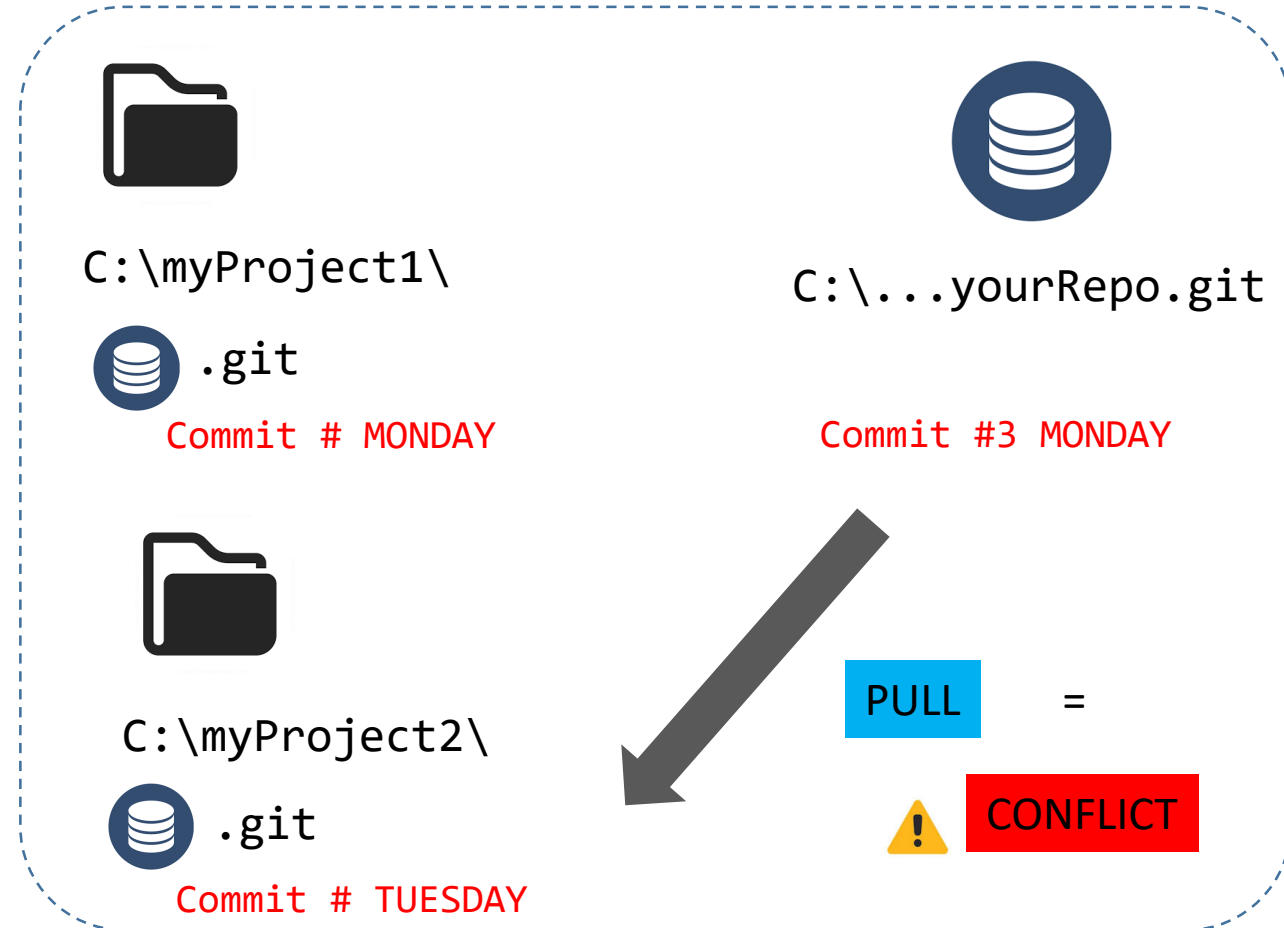
<h1>My app is **MONDAY** </h1>

2- From **project2** , commit

<h1>My app is **TUESDAY** </h1>

3- From **project2** , pull

✓ Check you get a conflict



# #17 Let 's resolve the conflict

## 1- Understand the conflict mode

THIS IS THE **CURRENT**  
VERSION

THIS IS THE VERSION  
FROM **OUTSIDE**

```
<<<<<< HEAD (Current Change)
|   <h1 class="title">My app is  TUESDAY</h1>
|
|=====
|   <h1 class="title">My app is MONDAY</h1>
|>>>>>> 6516e964fc940f8cb758f581347cb872faa7fd1b (Incoming Change)
```

## 2- Click on “Accept current change”

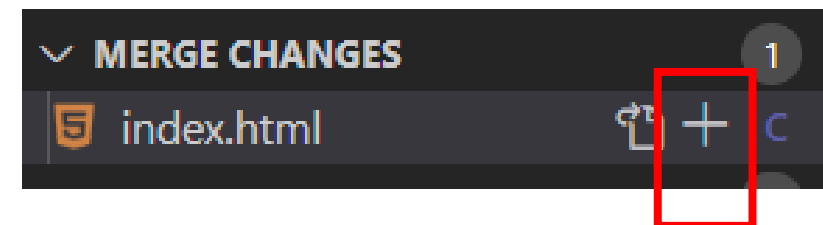
## 3- Change the title to

<h1>My app is **MONDAY OR TUESDAY** </h1>

## 4 – Stage change !!

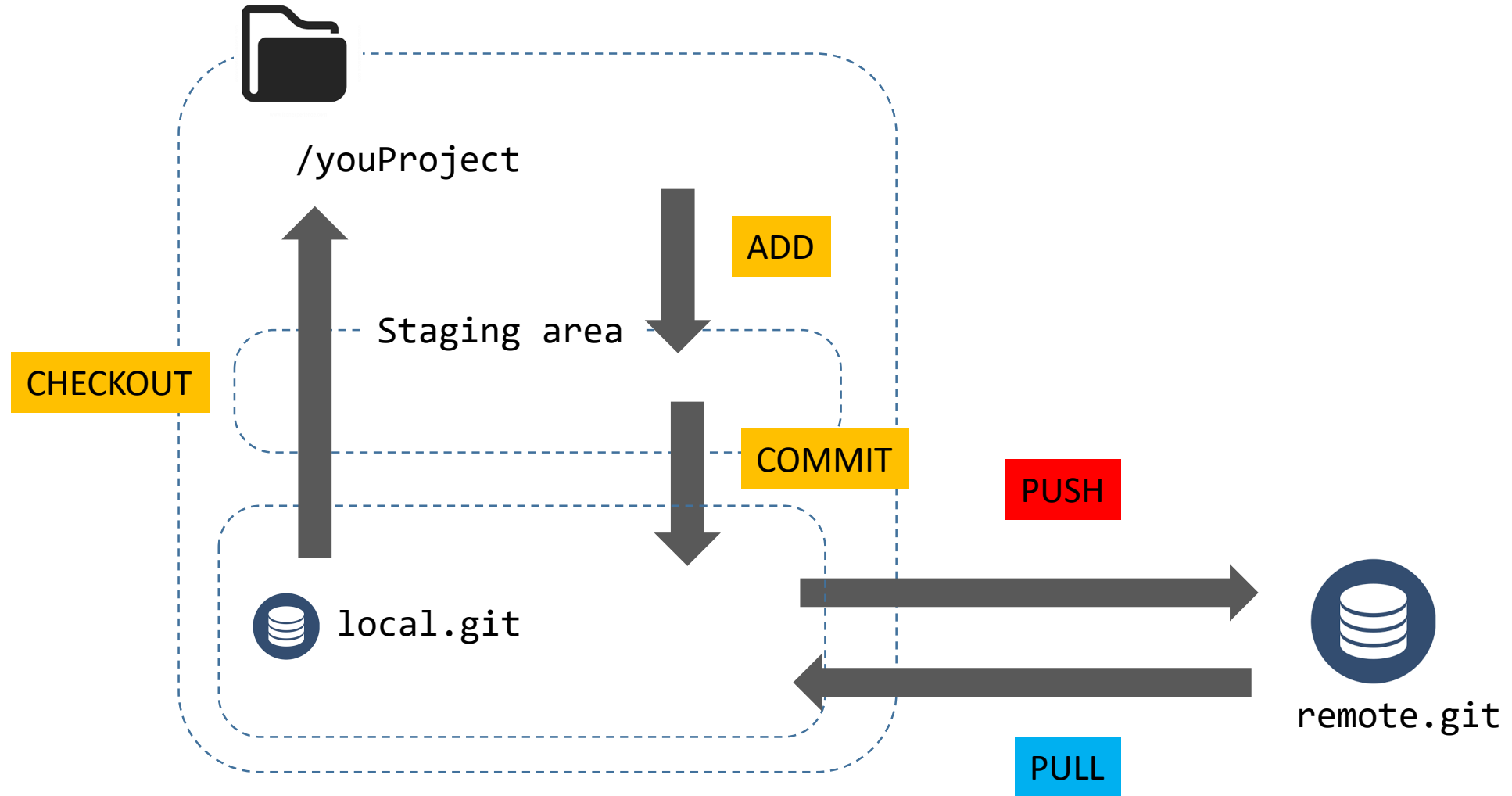
## 5 – Commit and push

✓ Pull on project 1 , and check this merge commit has been added



*Stage the change (i.e. GIT add), before committing !*

# To Sum up... all of this on MY computer

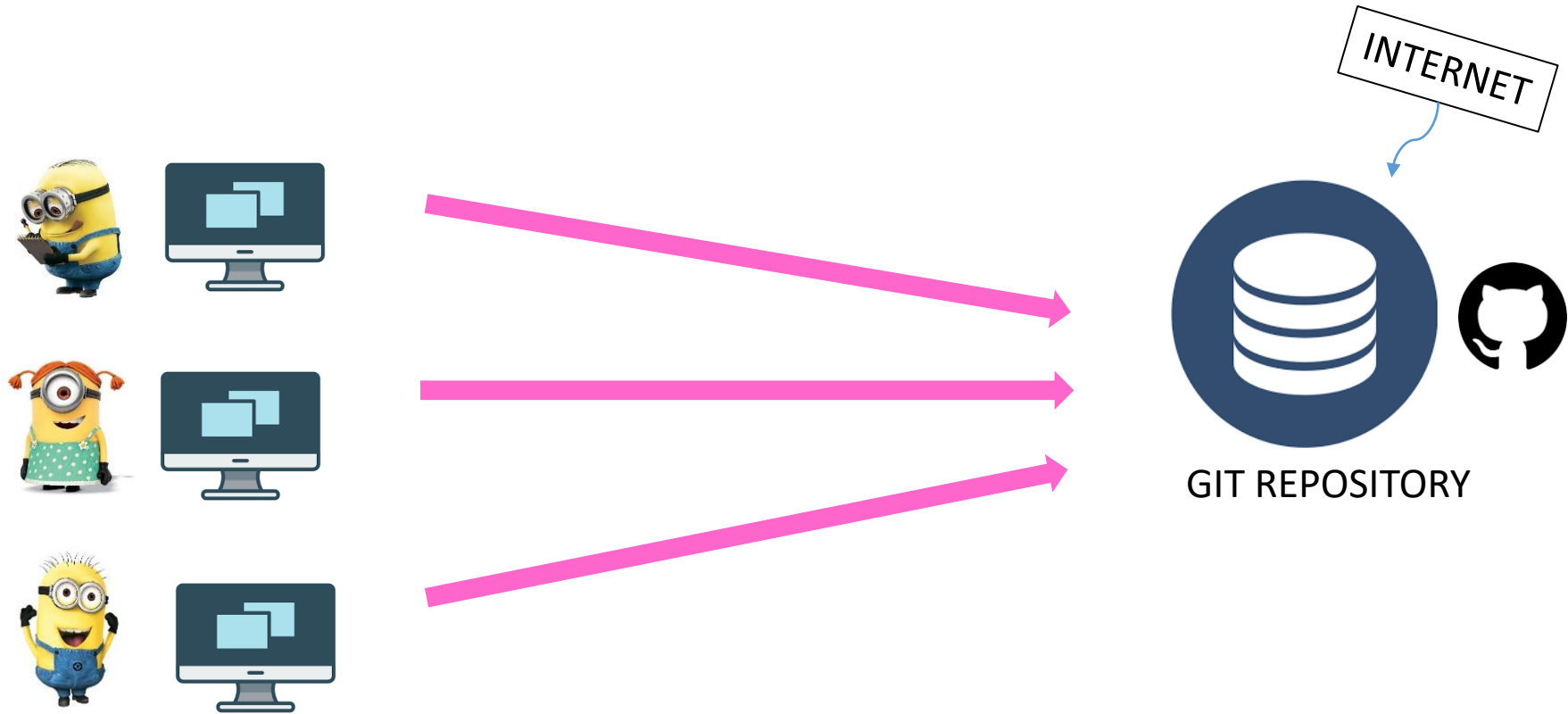






**DO IT !**

# OK, SO NOW LETs WORK WITH OTHER PEOPLE !



## #18 Clone from GitHub

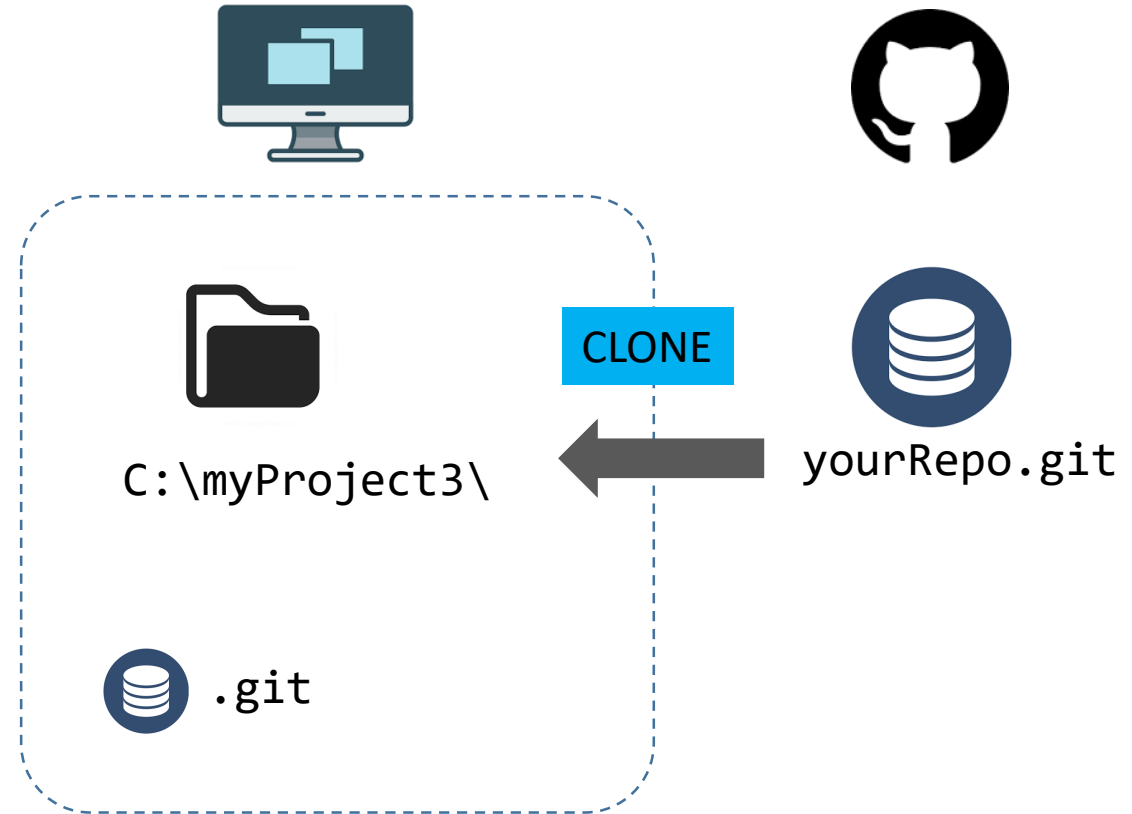
1 – Create a project on GITHUB

2 – Create a folder myProject3

3 – Clone the GitHub project

```
git clone https://github.com/xx.git
```

✓ Check an folder is created with a .git folder



## #18-1 Setup your global GIT configuration

1 – Run the below 2 commands:

```
git config --global user.name "YourFirstName YOurLastName"  
git config --global user.email YourMail
```

✓ Check the bellow file has been updated with this information:

```
C:\Users\YourUserName\.gitconfig
```

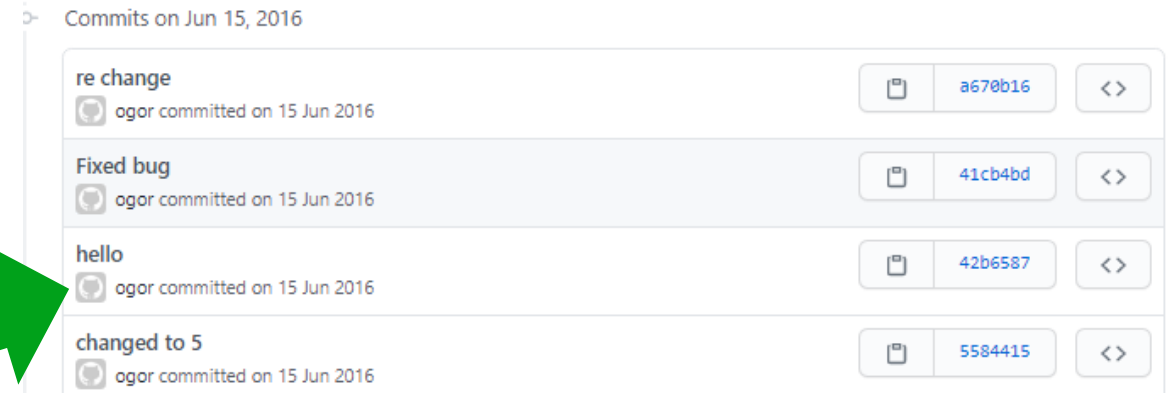
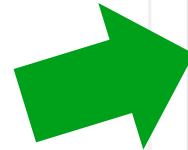
# #19 Check commits on GIT HUB

## 1 – Commit and push 4 changes (make 4 commit, not just 1) :

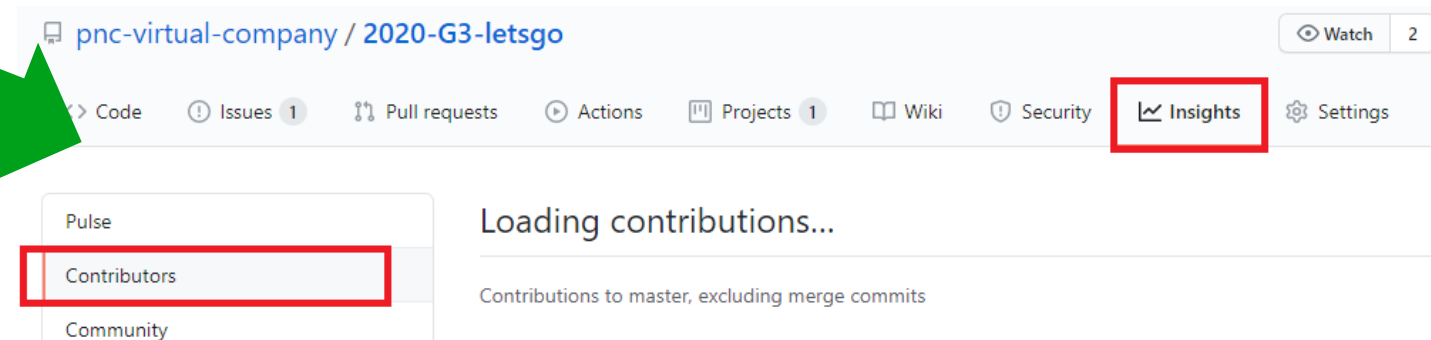
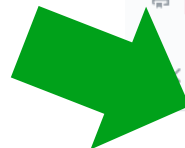
- Create an HTML file with a menu : home / Contact
- Create a CSS file to display menu in green
- Add an image below menu
- Add a H1 title



Check you can see the history of commit in GITHUB



Also check your commit is displayed in the contributions





**DO IT !**

#20 to #25

## TEAM OF 5 - WORK

LEADER OF  
THE WEEK



**WORK WITH YOUR VC2 GROUP !**

**COMMUNICATE IN YOUR VC2 SKYPE GROUP**

## #20 No conflicts Preparation...

(LEADER ONLY)

- Create a repository
- Invite other team members to your project and send them the GIT URL
- Determine who will be the student 1, 2, 3, 4, 5



STUDENT 1



STUDENT 2



STUDENT 3



STUDENT 4



STUDENT 5



(ALL)

- Clone this project to your computer (folder /projectTeam/)



## #20 No conflicts

1- Push 5 commits (1 per student) :

### Student 1

- Update menu
  - Home
  - Contact
  - Apply job
  - Help

### Student 2

- Add header title "PNC website"
- Add a paragraph  
Welcome to PNC !

### Student 3

Create Help page  
containing a H1

`<H1>Help</H1>`

### Student 4

Create Contact page  
containing a H1

`<H1>Contact</H1>`

### Student 5

Create Apply page  
containing a H1

`<H1>Apply</H1>`

2- Wait for everyone to push their changes

3- Push again 5 commits (1 per student)

### Student 1

- Link menu  
Contact to  
contact page

### Student 2

- Link menu Apply  
job to Apply page

### Student 3

Link menu Help  
to Help page

### Student 4

CSS :  
Set all H1 in RED

### Student 5

CSS :  
Set Menu in BLUE



Try to understand after EACH pull, what are the changes, by comparing the commits and changes



**DO IT !**

# WARNING BEFORE STARTING

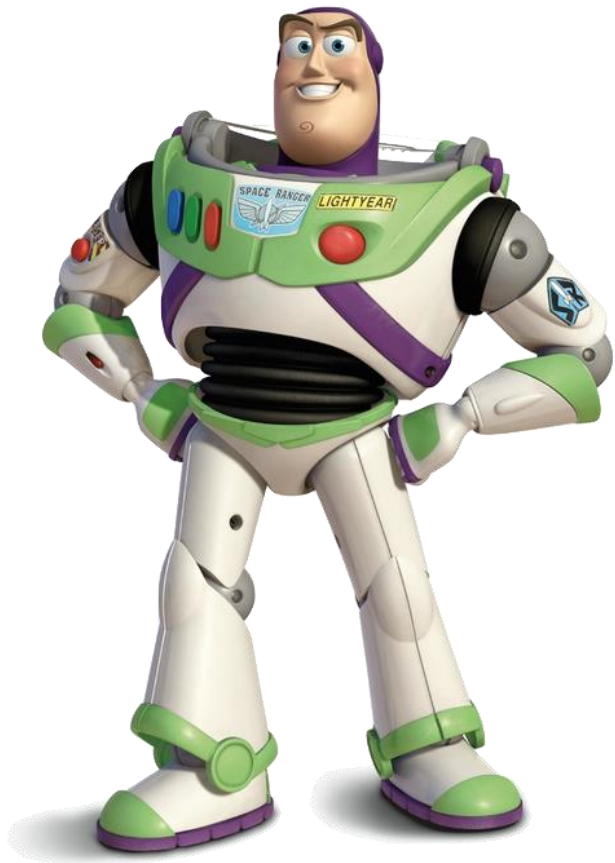
**YOU MUST DO ONLY THE WORK ASSIGNED TO YOU !**

(student 1 , student 2 etc....) and let other people push their changes

**DON'T PULL BEFORE STARTING**

First commit your changes  
Then fetch (refresh) and pull

BECAUSE WE WANT  
TO GET A CONFLICT !



## #21 Conflicts

(LEADER ONLY)

Create a new GITHUB PROJECT (**toyStory**)

Clone it, and push the following content [\(click me\)](#)

Invite other team members to this project and send them the GIT URL

(ALL)

- Clone this project to your computer (folder /projectToyStory/)



Everyone should clone and see the starting project



Item 1

**title of the page**



Card comment

# #21

## STUDENT 1 - ONLY



- **On HTML file:**

- Add a new section under the menu, and above the title of the page :
- Add a image on this section

src = "images/movie.jpg"

- **On CSS file :**

- Create a new CSS class "header-image" to define the style of this section
  - top/bottom margins to 50 pixels

# #21

## STUDENT 2 - ONLY



- **On HTML file:**

- Add a new card "Buzz"

|       |   |                             |
|-------|---|-----------------------------|
| image | = | images/buzz.png             |
| text  | = | Buzz is the friend of woody |

- Update the title of the page to integrate Buzz : "Toy Story world contains the characters ... and Buzz"
  - Add a menu item to "Buzz life" and link to :  
[https://en.wikipedia.org/wiki/Buzz\\_Lightyear](https://en.wikipedia.org/wiki/Buzz_Lightyear)

# #21

## STUDENT 3 - ONLY



### - On HTML file:

- Add a new section under the menu, and above the title of the page :
- the text to display is the following :

Toy Story is a Disney media franchise that commenced in 1995 with the release of the animated feature film of the same name, produced by Pixar Animation Studios and released by Walt Disney Pictures. The franchise is based on the anthropomorphic concept that all toys, unknown to humans, are secretly alive, and the films focus on a diverse group of toys that feature a classic cowboy doll named Sheriff Woody and a modern spaceman action figure named Buzz Lightyear,

### - On CSS file :

- create a new CSS class "header-description" to define the style of this section
- Set this section left/right margins to 100 pixels

#21

## STUDENT 4 - ONLY



### - On HTML file:

- Update the text of the card "woody"  
text = Woody is the main character
- Set the title of the page : "Toy Story world contains the character Woody"
- Set the first menu item to "Woody life" and link to :  
[https://en.wikipedia.org/wiki/Sheriff\\_Woody](https://en.wikipedia.org/wiki/Sheriff_Woody)
- Set the HTML page title to "Toy Story World !"



# #21

## STUDENT 5 - ONLY



### - On HTML file:

- Add a new card "Jessie"

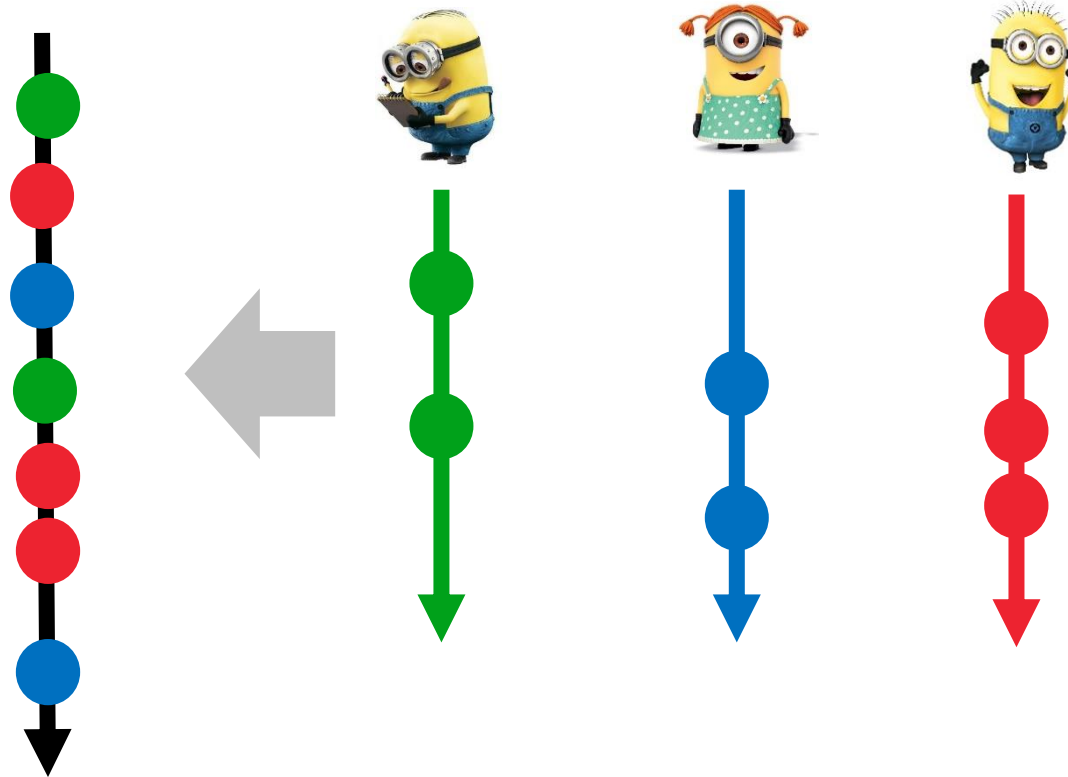
|       |   |                         |
|-------|---|-------------------------|
| image | = | images/jessie.png       |
| text  | = | Jessie is a good friend |

- Update the title of the page to integrate Jessie : "Toy Story world contains the characters ... and Jessie"
- Add a menu item to "Jessie life" and link to :  
[https://en.wikipedia.org/wiki/Jessie\\_\(Toy\\_Story\)](https://en.wikipedia.org/wiki/Jessie_(Toy_Story))

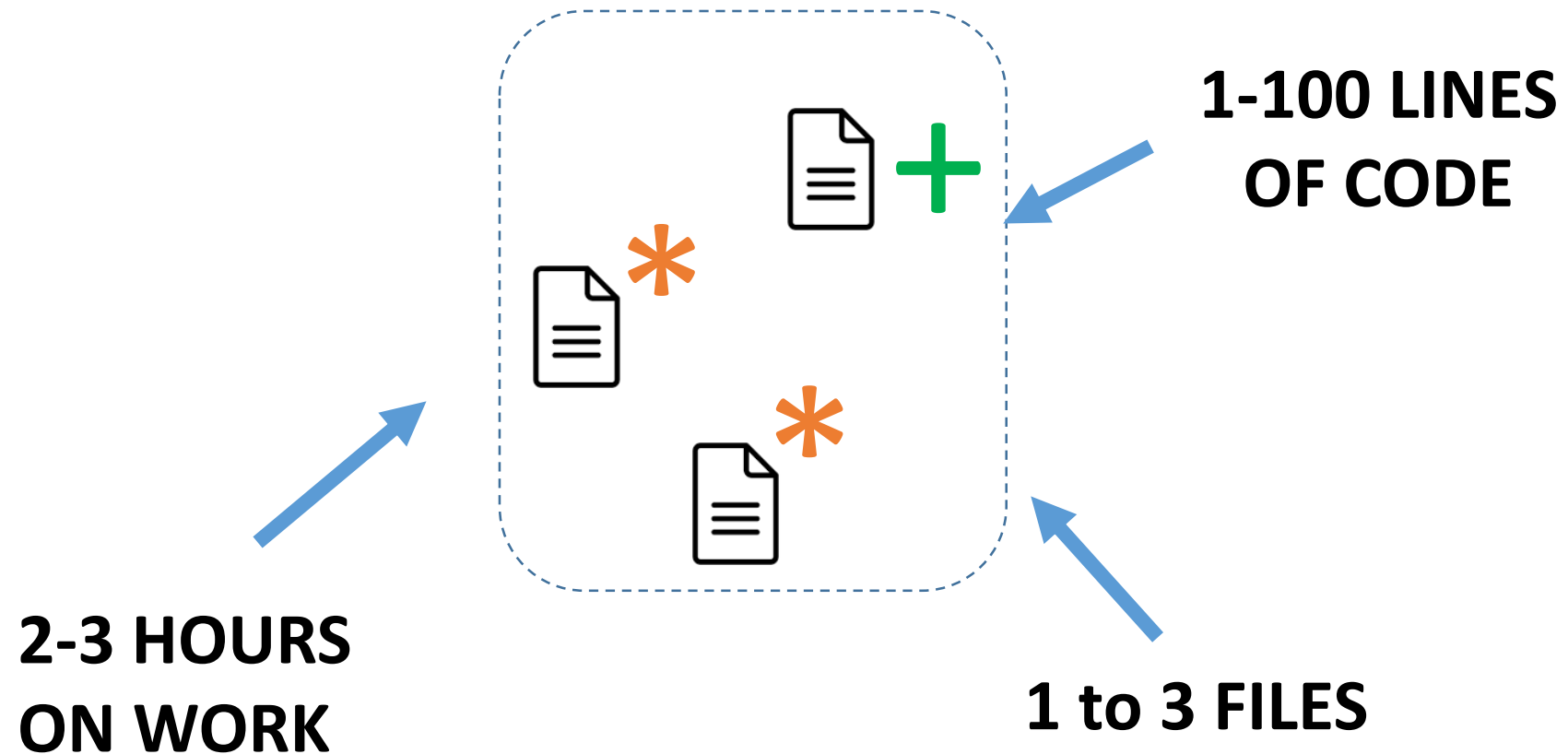


**DO IT !**

# WHY BRANCHING ?



# A COMMIT IS A SMALL CHANGE



# A FEATURE CAN REQUIRES MANY COMMITS



GITHUB  
ISSUE ID



#145 : Refactoring on class name

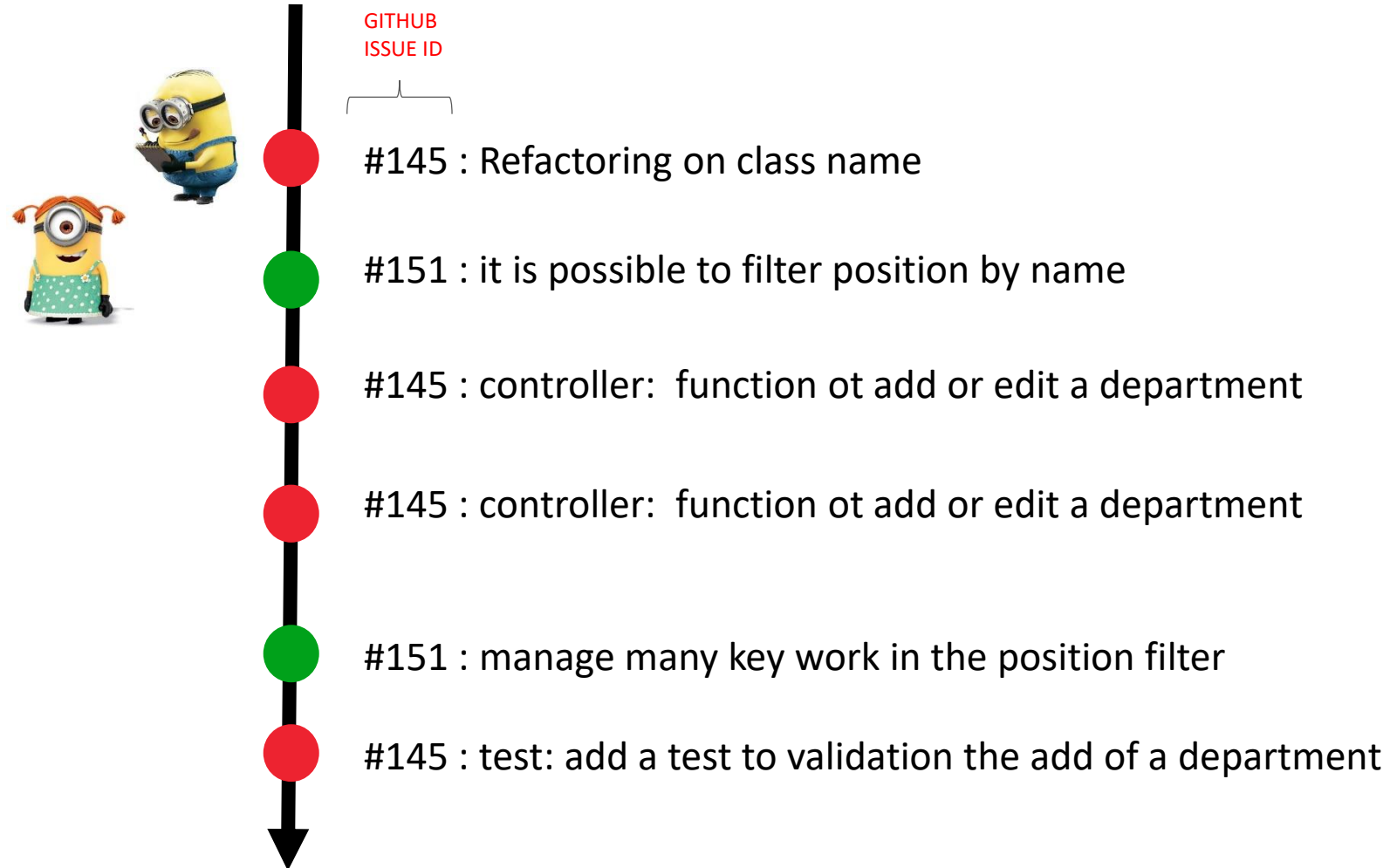
#145 : controller: function ot add or edit a department

#145 : view : add/edit department pop-up

#145 : test: add a test to validation the add of a department

# COMMITTING in 1 BRANCH CAN BE A MESS

WHEN PEOPLE ARE WORKING ON DIFFERENT FEATURES



# WE USE BRANCHES TO DEVELOPP IN PARALLEL

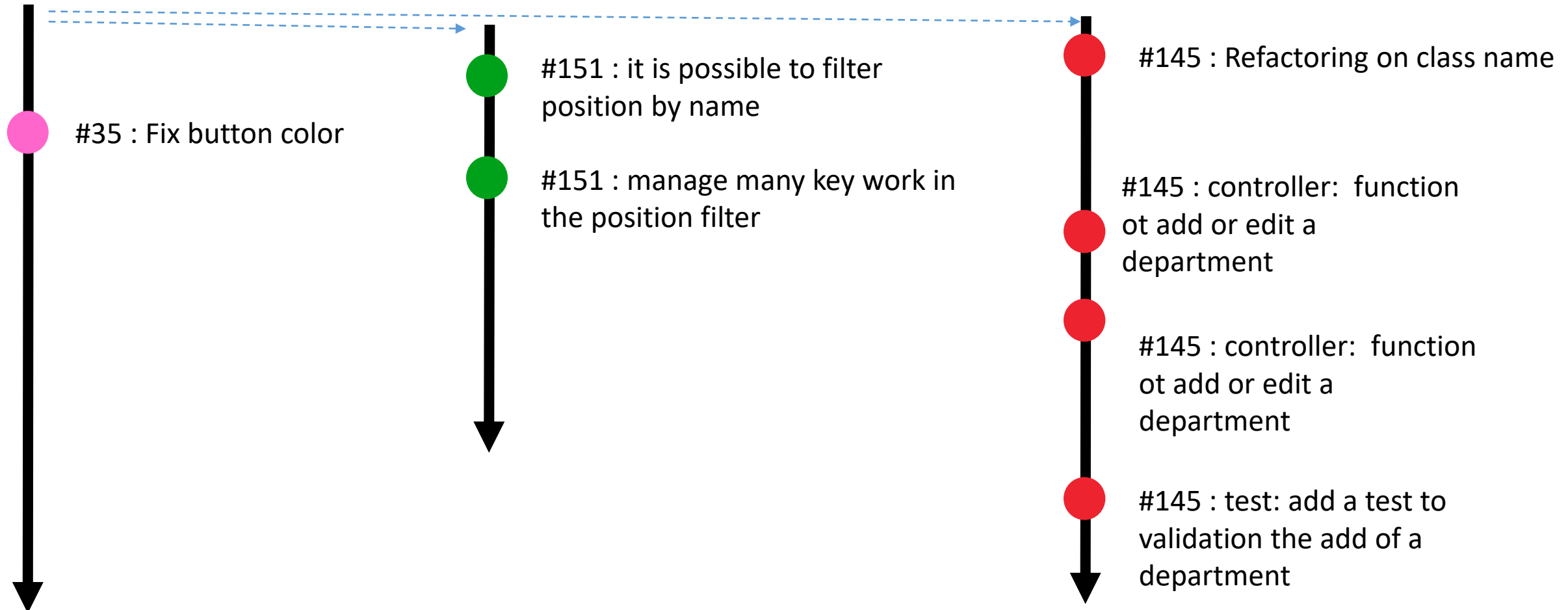


# 151 position view

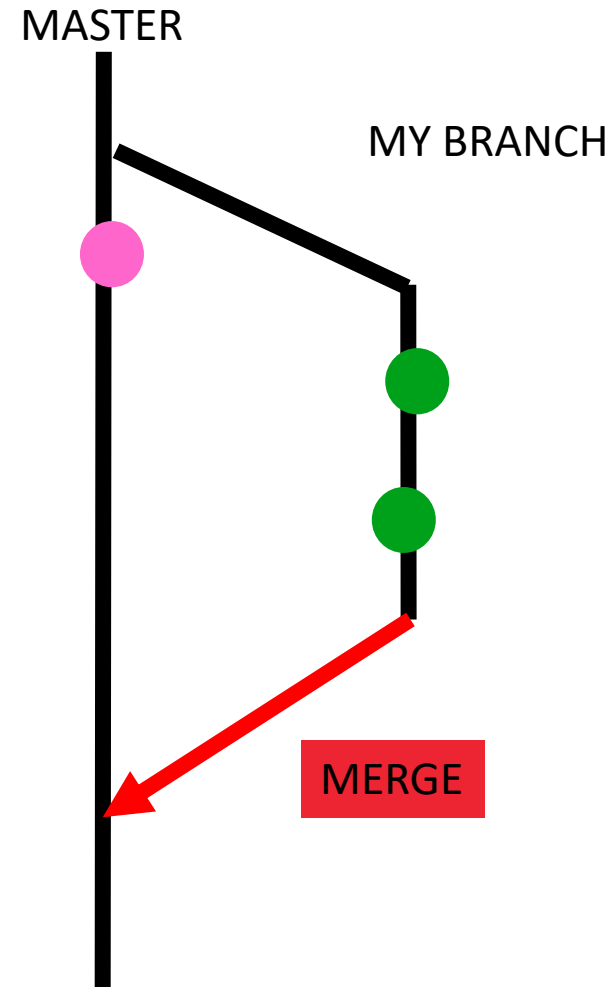


#145 department view

MASTER



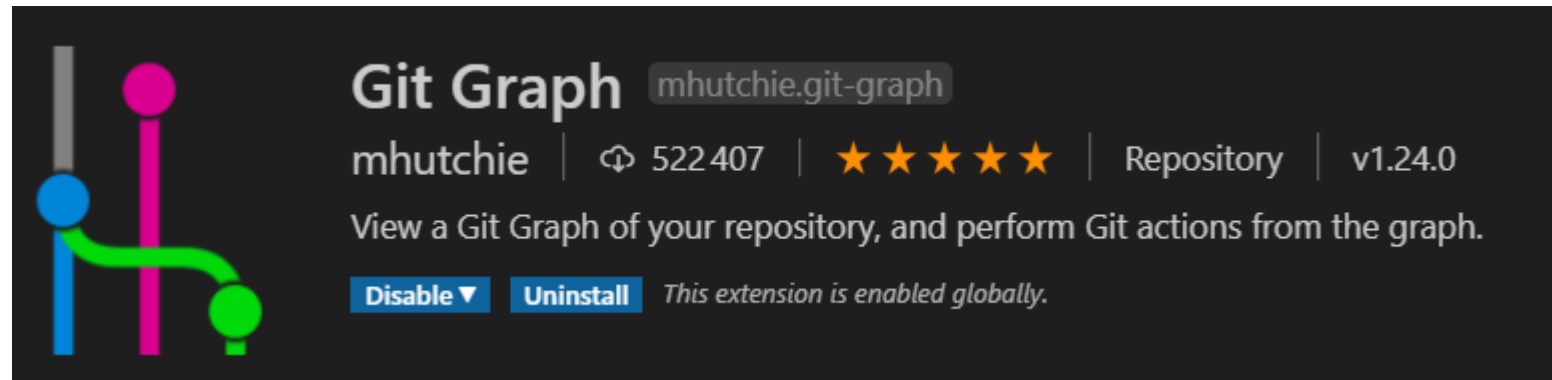
# BRING BACK A BRANCH = MERGE



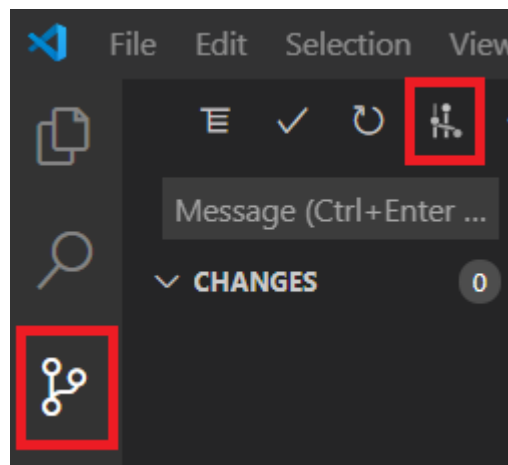


# Before starting...

Install GIT GRAPH extension on VS Code



✓ View the graph : check there is only one branch right now



## #22 Let's branch (individual work)

Create a new GitHub project (testBranches) and clone it

Commit and push 1 change

- Create an HTML file with a H1

Create branch menuBranch by running the checkout command with the -b switch:

```
git checkout -b "menuBranch "
```

From this branch, commit and push 1 change

- Add a menu to your HTML file

✓ Check your commit in on branch menuBranch but not on branch master

✓ Check the branch tree using GIT graph

✓ Check the branch content from GIT HUB

## #23 Let's merge (the command line way)

## #24 Let's merge (the VS code way)

## #25 Let's merge (the GITHUB way)