**GIT**

[**https://git-scm.com/book/es/v1/Fundamentos-de-Git-Obteniendo-un-repositorio-Git**](https://git-scm.com/book/es/v1/Fundamentos-de-Git-Obteniendo-un-repositorio-Git)

[**https://try.github.io/levels/1/challenges/1**](https://try.github.io/levels/1/challenges/1)

[**Inicializando un repositorio en un directorio existente**](https://git-scm.com/book/es/v1/Fundamentos-de-Git-Obteniendo-un-repositorio-Git#Inicializando-un-repositorio-en-un-directorio-existente)

Si estás empezando el seguimiento en Git de un proyecto existente, necesitas ir al directorio del proyecto, abrir la consola de git allí, (botón contextual Git Bash Here)

Esto crea un nuevo subdirectorio llamado .git que contiene todos los archivos necesarios del repositorio —un esqueleto de un repositorio Git.

git init

iprado1@pc-iprado1 MINGW64 /d/DESARROLLO/GIT

$ git init

Initialized empty Git repository in D:/DESARROLLO/GIT/.git/

Si hay archivos en la carpeta donde se creó el repositorio al consultar el Status, lo que hace es marcarlos como untracked"

$ git status

On branch master

Initial commit

Untracked files:

(use "git add <file>..." to include in what will be committed)

test.html

testGit.js

nothing added to commit but untracked files present (use "git add" to track)

Para añadir por ejemplo el archivo testGit.js

$ git add testGit.js

Si ahora volvemos a ver el status:

$ git status

On branch master

Initial commit

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: testGit.js

Untracked files:

(use "git add <file>..." to include in what will be committed)

test.html

Me indica que hay un archivo pendiente de commit, el que acabo de añadir.

Notice how Git says changes to be committed? The files listed here are in the Staging Area, and they are not in our repository yet. We could add or remove files from the stage before we store them in the repository.

Para subirlo escribimos: **git commit –m “comentarios”;**

$ git commit -m "Add testGit.js"

[master (root-commit) 09da44a] Add testGit.js

1 file changed, 7 insertions(+)

create mode 100644 testGit.js

Para subir varios js a la vez, haríamos **> git add '\*js'**

Subo el otro archivo que tengo en el directorio que no he añadido antes, con add, hago el commit y al consultar ahora el status tengo lo siguiente:

$ git status

On branch master

nothing to commit, working tree clean

No tengo cambios pendientes de subir!

Para consultar el registro de cambios tenemos el log: **git log**

$ git log

commit 4b645384d637ae13bfa847e2787ceef3393f46fd

Author: inmaprado <inmaprado@gmail.com>

Date: Mon Jan 9 14:55:04 2017 +0100

Add test.html

commit 09da44aae4d0c7fedc9bdd2860f0e3bc2e9da6b5

Author: inmaprado <inmaprado@gmail.com>

Date: Mon Jan 9 14:51:12 2017 +0100

Add testGit.js

Desde la plataforma github, en mi cuenta creo un nuevo repositorio <https://github.com/inmaprado/Test.git>

Para conectarme con el uso la siguiente expresión:

$ git remote add origin <https://github.com/inmaprado/Test.git>

The push command tells Git where to put our commits when we're ready, and boy we're ready. So let's push our local changes to our **origin** repo (on GitHub).

The name of our remote is origin and the default local branch name is master. The -u tells Git to remember the parameters, so that next time we can simply run git push and Git will know what to do. Go ahead and push it!

$ git push -u origin master

Counting objects: 6, done.

Delta compression using up to 4 threads.

Compressing objects: 100% (5/5), done.

Writing objects: 100% (6/6), 659 bytes | 0 bytes/s, done.

Total 6 (delta 0), reused 0 (delta 0)

To https://github.com/inmaprado/Test.git

\* [new branch] master -> master

Branch master set up to track remote branch master from origin.

(Me pide usuario y password para conectarse a github )

Imagina que haces unos cambios en uno de los archivos que tengo en el repo, por ejemplo en testGit.js, si ahora ejecutamos los siguiente, git diff HEAD me mostrará los cambios realizados

$ git diff HEAD

**diff --git a/testGit.js b/testGit.js**

**index 77ad4b1..2556990 100644**

**--- a/testGit.js**

**+++ b/testGit.js**

@@ -1,7 +1,6 @@

function sayHi(name){

-

return "Hi " + name +"!!!!";

}

-console.log("--------------------")

-console.log(sayHi("inma"));

-console.log("--------------------")

+

+console.log(sayHi("gitHub"));

+

iprado1@pc-iprado1 MINGW64 /d/DESARROLLO/GIT (master)

Con – me indica el código eliminado y + en verde lo añadido

The--staged option to see the changes you just staged.

$ git add testGit.txt

iprado1@pc-iprado1 MINGW64 /d/DESARROLLO/GIT (master)

$ git diff --staged

**diff --git a/testGit.txt b/testGit.txt**

**new file mode 100644**

**index 0000000..e69de29**

You can unstage files by using the git reset command.

$ git reset testGit.txt

Unstaged changes after reset:

M testGit.js

Files can be changed back to how they were at the last commit by using the command: git checkout -- <target>.

$ git checkout -- testGit.js

When developers are working on a feature or bug they'll often create a copy (aka. branch) of their code they can make separate commits to

Ahora si escribimos $ git branch vemos las dos ramas

$ git branch

limpieza

\* master

You can switch branches using the git checkout <branch> command

Para cambiarme de rama:

iprado1@pc-iprado1 MINGW64 /d/DESARROLLO/GIT (master)

$ git checkout limpieza

Switched to branch 'limpieza'

Ahora vemos en verde la rama limpieza que es donde me encuentro.

$ git branch

\* limpieza

master

Para eliminar un fichero, en este caso todos los ficheros .txt

$ git rm '\*.txt'

rm 'testGit.txt'

Despues de eliminar hago un commit, y tengo lo siguiente:

$ git commit -m "removing the txt"

[limpieza 08ae3a1] removing the txt

1 file changed, 1 deletion(-)

delete mode 100644 testGit.txt

Vuelvo a la rama master

iprado1@pc-iprado1 MINGW64 /d/DESARROLLO/GIT (limpieza)

$ git checkout master

Switched to branch 'master'

Your branch is up-to-date with 'origin/master'.

Ahora para realizar un merge entre las dos ramas:

Alrighty, the moment has come when you have to merge your changes from the limpieza  branch into the master branch

$ git merge limpieza

Updating 4b64538..08ae3a1

Fast-forward

Eliminar una rama:

You can use git branch -d <branch name> to delete a branch.

$ git branch -d limpieza

Deleted branch limpieza (was 08ae3a1).

Para subir al repositorio:

$ git push

Counting objects: 4, done.

Delta compression using up to 4 threads.

Compressing objects: 100% (3/3), done.

Writing objects: 100% (4/4), 451 bytes | 0 bytes/s, done.

Total 4 (delta 0), reused 0 (delta 0)

To https://github.com/inmaprado/Test.git

4b64538..08ae3a1 master -> master