

Computación en Física

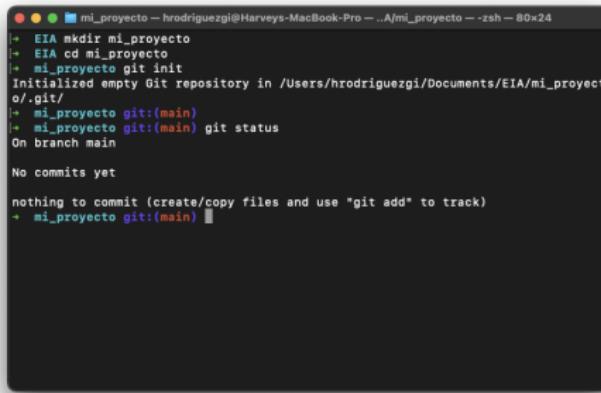
Harvey Rodriguez Gil

Universidad EIA

22 de Agosto de 2024

Inicializando un Repositorio de GIT - Local

- ▶ Creamos la carpeta: `mkdir mi_proyecto`
- ▶ Ingresamos a la carpeta: `cd mi_proyecto`
- ▶ Inicializamos el repositorio: `git init`
- ▶ Validamos el estado del repositorio: `git status`



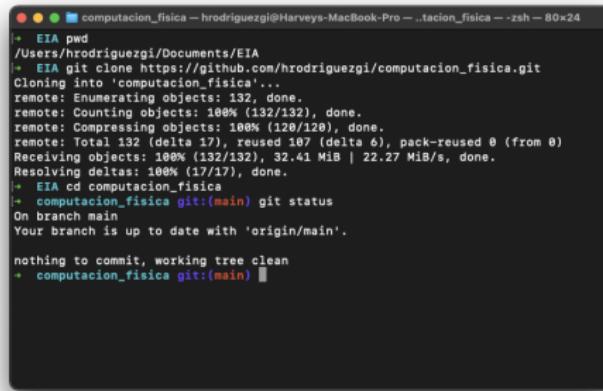
```
mi_proyecto ~ hrodriguezg@Harveys-MacBook-Pro ~ ..A/mi_proyecto ~ -zsh - 80x24
+- EIA mkdir mi_proyecto
+- EIA cd mi_proyecto
+- mi_proyecto git init
Initializing empty Git repository in /Users/hrodriguezg/Documents/EIA/mi_proyecto/.git/
+- mi_proyecto git:(main)
+- mi_proyecto git:(main) git status
On branch main

No commits yet

nothing to commit (create/copy files and use "git add" to track)
+- mi_proyecto git:(main) █
```

Inicializando un Repositorio de GIT - Remoto

- ▶ Nos ubicamos en una carpeta donde clonaremos el repositorio
- ▶ Clonamos el repositorio a partir de la url: git clone
`https://github.com/hrodriguezgi/computacion_fisica.git`
- ▶ Ingresamos a la carpeta donde se ha clonado el repositorio: cd
`computacion_fisica`

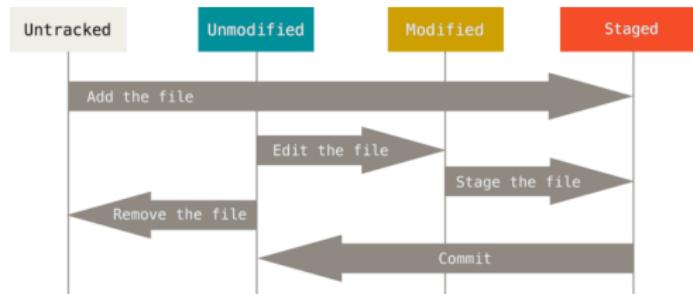


```
computacion_fisica ~ hrodriguezgi@Harveys-MacBook-Pro ~ ..tacion_fisica ~ zsh ~ 80x24
+- EIA pwd
/Users/hrodriguezgi/Documents/EIA
+- EIA git clone https://github.com/hrodriguezgi/computacion_fisica.git
Cloning into 'computacion_fisica'...
remote: Enumerating objects: 132, done.
remote: Counting objects: 100% (132/132), done.
remote: Compressing objects: 100% (120/120), done.
remote: Total 132 (delta 17), reused 107 (delta 6), pack-reused 0 (from 0)
Receiving objects: 100% (132/132), 32.41 MiB | 22.27 MiB/s, done.
Resolving deltas: 100% (17/17), done.
+- EIA cd computacion_fisica
+- computacion_fisica git:(main) git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
+- computacion_fisica git:(main)
```

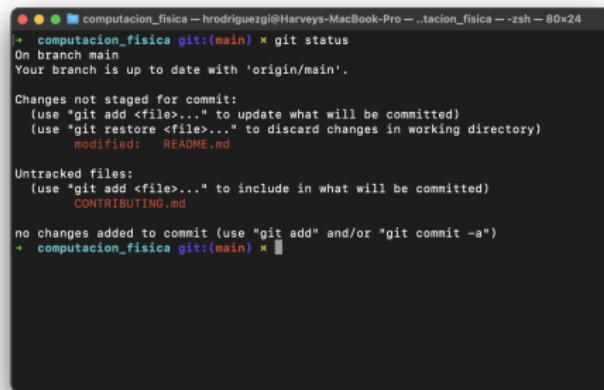
Ciclo de vida

En GIT los archivos son parte de uno de los dos grupos: tracked o untracked. Los primeros son todos aquellos de los que GIT tiene conocimiento, y pueden estar entre unmodified, modified y staged.



Ciclo de vida

Cuando modificamos archivos existentes estos se encuentran en tracked bajo el estado modified, sin embargo cuando agregamos un archivo nuevo, su estado inicial será untracked



```
+ computacion_fisica git:(main) ✘ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified: README.md

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        CONTRIBUTING.md

no changes added to commit (use "git add" and/or "git commit -a")
+ computacion_fisica git:(main) ✘
```

Ciclo de vida

Cuando estos archivos **untracked** son adicionados a git por medio del `git add`, los siguientes cambios los veremos simplemente como de modificaciones más no como de archivos que no están en el índice de GIT

```
computacion_fisica — hrodriguezgi@Harveys-MacBook-Pro — ..tacion_fisica — zsh — 80x24
+ computacion_fisica git:(main) ✘ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   CONTRIBUTING.md
    modified:   README.md

+- computacion_fisica git:(main) ✘ vim CONTRIBUTING.md
+- computacion_fisica git:(main) ✘ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   CONTRIBUTING.md
    modified:   README.md

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   CONTRIBUTING.md
```

Ignorando Archivos

GIT nos permite tener ficheros en nuestra carpeta donde se encuentra el repositorio sin que esto sea registrado en el indice del mismo, ya sea porque son ficheros de logs, o creados por el sistema de forma automática, entre otros. Para esto se crea un archivo llamado `.gitignore` y se define lo que se quiere excluir

```
● ● ● computacion_fisica - hrodriguezgil@Harveys-MacBook-Pro - ..tacion_fisica -- zsh - 80x24
-> computacion_fisica git:(main) ✘ cat .gitignore
*.csv
-> computacion_fisica git:(main) ✘ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   CONTRIBUTING.md
    modified:   README.md

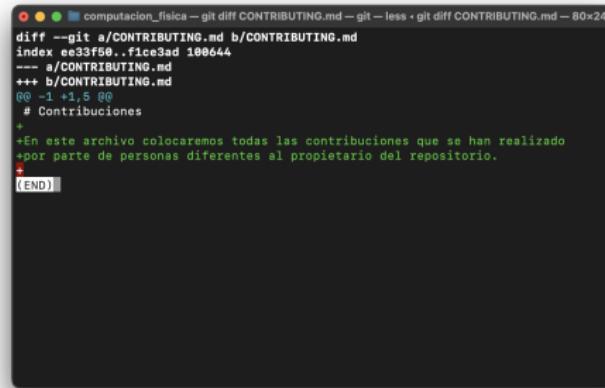
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   CONTRIBUTING.md

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

-> computacion_fisica git:(main) ✘ ls
CONTRIBUTING.md  clase_1      clase_3      clase_6      clase_9
FILE.csv         clase_10     clase_4      clase_7      images
README.md        clase_2      clase_5      clase_8
```

Visualizando los cambios

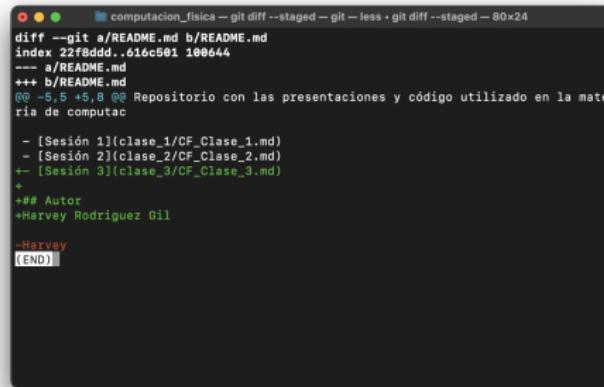
Cuando hacemos un cambio sobre uno o más ficheros de nuestro repositorio, el comando `git status` puede quedarse un poco corto con el detalle de los cambios, por lo que existe otro comando llamado `git diff` que nos permite visualizar de forma más explícita las modificaciones:



```
computacion_fisica -- git diff CONTRIBUTING.md -- git - less + git diff CONTRIBUTING.md -- 80x24
diff --git a/CONTRIBUTING.md b/CONTRIBUTING.md
index ee33f50..f1ce3ad 100644
--- a/CONTRIBUTING.md
+++ b/CONTRIBUTING.md
@@ -1 +1,5 @@
 # Contribuciones
+
+En este archivo colocaremos todas las contribuciones que se han realizado
+por parte de personas diferentes al propietario del repositorio.
+
```

Visualizando los cambios

Cuando los archivos a los que queremos identificar los cambios se encuentran en el área de stage, el comando a ejecutar será con el argumento de `--staged`



```
computacion_fisica - git diff --staged - git - less + git diff --staged - 80x24
diff --git a/README.md b/README.md
index 22f8bdd..616c501 100644
--- a/README.md
+++ b/README.md
@@ -5,5 +5,8 @@ Repositorio con las presentaciones y código utilizado en la mate
ria de computac

- [Sesión 1](clase_1/CF_Clase_1.md)
- [Sesión 2](clase_2/CF_Clase_2.md)
+- [Sesión 3](clase_3/CF_Clase_3.md)
+
+## Autor
+Harvey Rodriguez Gil

-Harvey
(END)
```

Persistiendo cambios

Para persistir los cambios en GIT hacemos uso del comando `git commit`, el cual si lo ejecutamos tal cual, nos abrirá un editor de texto donde debemos colocar el mensaje del commit, o una alternativa sería ejecutar `git commit -m "Updated README.md"`

The screenshot shows a terminal window with the following session:

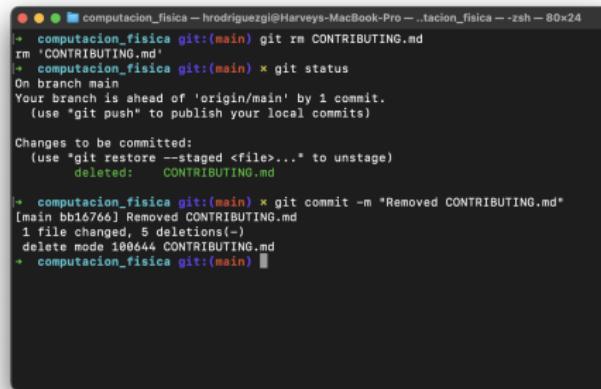
```
mi_proyecto git:(main) ✘ git status
On branch main
No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file: README.md

mi_proyecto git:(main) ✘ git commit -m "Updated README.md"
[master (root-commit) 80075c4] Updated README.md
 1 file changed, 3 insertions(+)
  create mode 100644 README.md
mi_proyecto git:(main) ✘ git status
On branch main
nothing to commit, working tree clean
mi_proyecto git:(main) ✘
```

Removiendo archivos

Para remover archivos debemos garantizar que estos no estén registrados en el índice, para esto ejecutaremos git rm



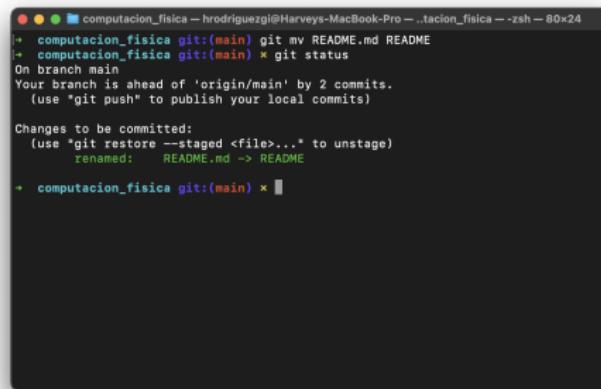
```
computacion_fisica - hrodriguezg@Harveys-MacBook-Pro - ..tacion_fisica - zsh - 80x24
+- computacion_fisica git:(main) git rm CONTRIBUTING.md
rm 'CONTRIBUTING.md'
+- computacion_fisica git:(main) ✘ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    deleted:   CONTRIBUTING.md

+- computacion_fisica git:(main) ✘ git commit -m "Removed CONTRIBUTING.md"
[main bb1676d] Removed CONTRIBUTING.md
  1 file changed, 5 deletions(-)
  delete mode 100644 CONTRIBUTING.md
+- computacion_fisica git:(main) ✘
```

Moviendo archivos

Para mover (renombrar) archivos GIT nos ofrece el comando para esto: `git mv` el cual se encarga del cambio correcto en el indice, de lo contrario deberíamos ejecutar 3 comandos: renombrar (linux), remover el viejo y adicionar el nuevo.



A screenshot of a macOS terminal window titled "computacion_fisica". The window shows the following command and its output:

```
computacion_fisica git:(main) git mv README.md README
computacion_fisica git:(main) git status
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    renamed:   README.md -> README

computacion_fisica git:(main) 
```

Visualizando la historia

En el momento en que se desea visualizar la cantidad de cambios que se han realizado en un repositorio, GIT nos ofrece el comando `git log`:

```
computacion_fisica - git log --git - less + git log - 80x24
commit bb167668f5b318cae5c8f6fc2c81bc9ed50e5a8 (HEAD -> main)
Author: Harvey Rodriguez Gil <harvey.rodriguez@finkargo.com>
Date:   Wed Aug 21 19:43:59 2024 -0500

    Removed CONTRIBUTING.md

commit ef316c2e69285a0399c25bf8a60ce9abcc4abf89
Author: Harvey Rodriguez Gil <harvey.rodriguez@finkargo.com>
Date:   Wed Aug 21 19:42:35 2024 -0500

    Added CONTRIBUTING.md

commit da75f355d8c13b4647e836ac462c85f92a6f1f6e (origin/main, origin/HEAD)
Author: Harvey Rodriguez Gil <harvey.rodriguez@finkargo.com>
Date:   Tue Aug 20 07:20:59 2024 -0500

    Updated README.md

commit add0b6e8787481ef3cc1a4b993260cdf5a8eab9b
Author: Harvey Rodriguez Gil <harvey.rodriguez@finkargo.com>
Date:   Thu Aug 8 07:10:54 2024 -0500

    Added class 10
:
```

Visualizando la historia

Si queremos tener un mayor nivel de detalle, podemos acompañar el anterior comando con `-p` que presentará las diferencias de los archivos modificados y `-2` para que solo nos muestre los últimos 2 cambios (commits)

```
computacion_fisica - git log -p -2 - git - less .git log -p -2 - 80x24
commit bb167668f5b318cae5c8f6f2c28c1bc9ed58e5a8 (HEAD -> main)
Author: Harvey Rodriguez Gil <harvey.rodriguez@finkargo.com>
Date:   Wed Aug 21 19:43:59 2024 -0500

        Removed CONTRIBUTING.md

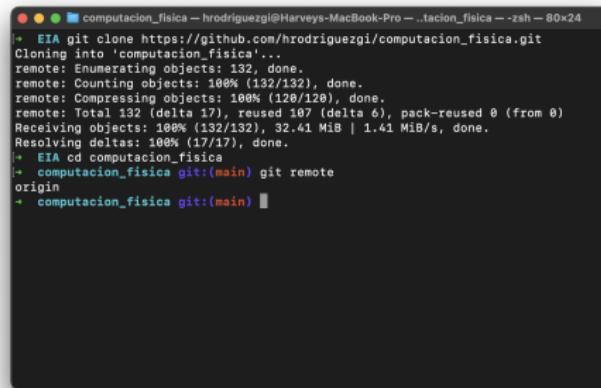
diff --git a/CONTRIBUTING.md b/CONTRIBUTING.md
deleted file mode 100644
index f1ce3ad..0000000
--- a/CONTRIBUTING.md
+++ /dev/null
@@ -1,5 +0,0 @@
-# Contribuciones
-
-En este archivo colocaremos todas las contribuciones que se han realizado
-por parte de personas diferentes al propietario del repositorio.
-


commit ef316c2e69285a0399c25bf8a60ce9abcc4abf89
Author: Harvey Rodriguez Gil <harvey.rodriguez@finkargo.com>
Date:   Wed Aug 21 19:42:35 2024 -0500

        Added CONTRIBUTING.md
```

Visualizando el repositorio remoto

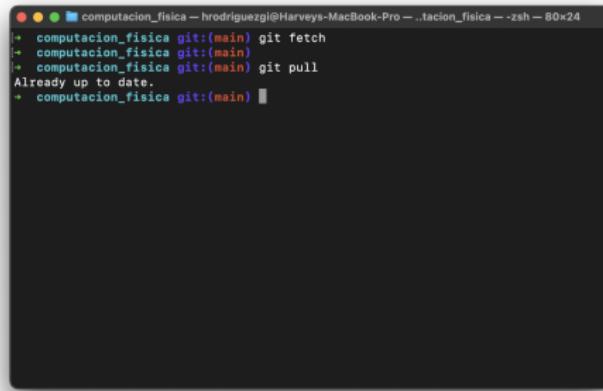
Una vez hemos clonado un repositorio remoto con ayuda del comando `git clone`, podemos ejecutar el comando `git remote` para validar que este luego de haber sido descargado en nuestro equipo local está conectado contra un servidor remoto



```
computacion_fisica - hrodriguezgi@Harveys-MacBook-Pro - ..tacion_fisica - -zsh - 80x24
* EIA git clone https://github.com/hrodriguezgi/computacion_fisica.git
Cloning into 'computacion_fisica'...
remote: Enumerating objects: 132, done.
remote: Counting objects: 100% (132/132), done.
remote: Compressing objects: 100% (120/120), done.
remote: Total 132 (delta 17), reused 107 (delta 6), pack-reused 0 (from 0)
Receiving objects: 100% (132/132), 32.41 MiB | 1.41 MiB/s, done.
Resolving deltas: 100% (17/17), done.
* EIA cd computacion_fisica
*+ computacion_fisica git:(main) git remote
origin
*+ computacion_fisica git:(main)
```

Actualizando el repositorio local

GIT cuenta con dos comandos para sincronizar el repositorio remoto en nuestros equipos, el primero es git fetch, el cual descarga los cambios que hayan sin afectar nuestra carpeta local, y el segundo es git pull el cual descarga y mezcla los cambios con nuestros archivos



```
computacion_fisica ~ hrodriguezgi@Harveys-MacBook-Pro ~ .tacion_fisica ~ zsh ~ 80x24
↳ computacion_fisica git:(main) git fetch
↳ computacion_fisica git:(main)
↳ computacion_fisica git:(main) git pull
Already up to date.
↳ computacion_fisica git:(main)
```

Publicando los cambios

Cuando hacemos cambios en nuestro repositorio local y queremos publicar esto en el repositorio remoto, se deberá tener los cambios ya en stage (`git add`) y haberlos persistido (`git commit`) para lograr publicarlos `git push`. Esto solo funciona si se tienen permisos sobre el repositorio

```
computacion_fisica ~ hrodriguezgi@Harveys-MacBook-Pro ..tacion_fisica -- zsh - 80x24
* computacion_fisica git:(main) ✘ git add README.md
* computacion_fisica git:(main) ✘ git commit -m "Updated README.md"
[main ccbledc] Updated README.md
 1 file changed, 3 insertions(+), 1 deletion(-)
* computacion_fisica git:(main) ✘ git push origin main
Enumerating objects: 6, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 10 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 1.00 KiB | 1.00 MiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/hrodriguezgi/computacion_fisica.git
  da76f35..ccbledc main -> main
* computacion_fisica git:(main) ✘
```

Vamos a practicar

- ▶ Ingresar a <https://github.com/>
- ▶ Crear un repositorio vacío
- ▶ Practicar los comandos vistos:
 - ▶ Clonar el repositorio en local
 - ▶ Crear un archivo de README.md
 - ▶ Crear un archivo de .gitignore
 - ▶ Guardar los cambios
 - ▶ Publicar los cambios en el repo

