
Comandes terminal Windows

Cristina Roigé Gaja

Table of Contents

Que és git	1
Instal·lar GIT a Windows	1
Configurar git	2
Crear repositori	3
Clonar repositoris	5
Espai de treball a GIT	6
Comandes status add dif	8
fitxer gitignore	9
Comanda commit i versions	10
Comandes reset i checkout	13
Desfer un commit	14
Tornar a un commit anterior	16
Buscar un commit	18
Branch	20
Comanda stash	23
Comanda cherry-pick	26
Resolució de conflictes.	30
Repositoris remots I GitHub	32
Una eina de gestió gràfica de GIT: gitkraken	39

Que és git

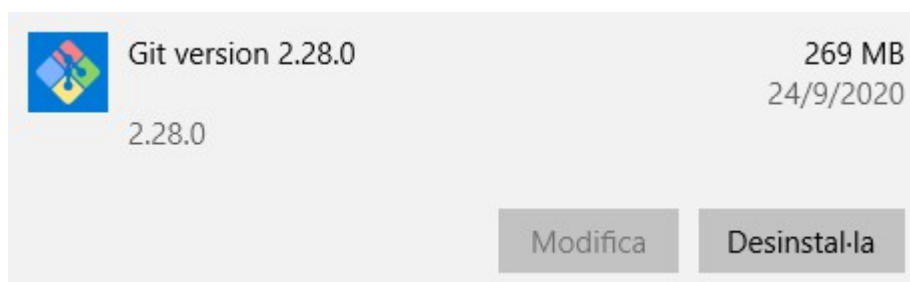
Git és un programa de control de versions que ha estat pensat per tindre una gran eficiència i confiabilitat del manteniment de versions de les nostres aplicacions.

Git l'utilitzarem per tindre control i registrar els canvis realitzats sobre un o diversos arxius de manera que si ens equivoquem o volem recuperar alguna versió antiga ho tinguem més fàcil que gestionant còpies de seguretat a pel.

Instal·lar GIT a Windows

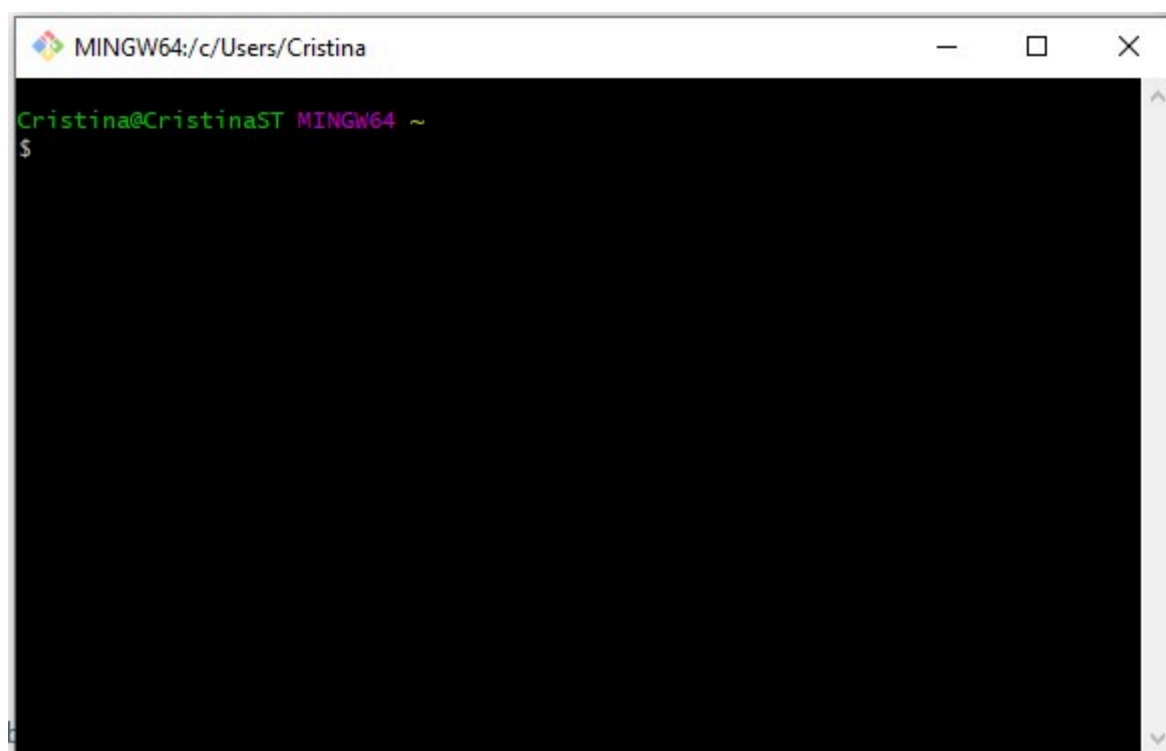
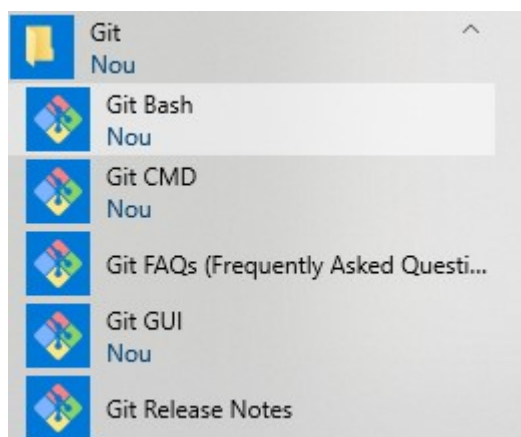
Per instal·lar GIT a windows una de les maneres que hi ha és anant a la següent pàgina <https://git-scm.com/> i descarregar la versió de Windows. un cop descarregat a baixades executarem el fitxer i el configurarem com vulguem.

Un cop instal·lat podem veure la versió que tinguem instal·lada

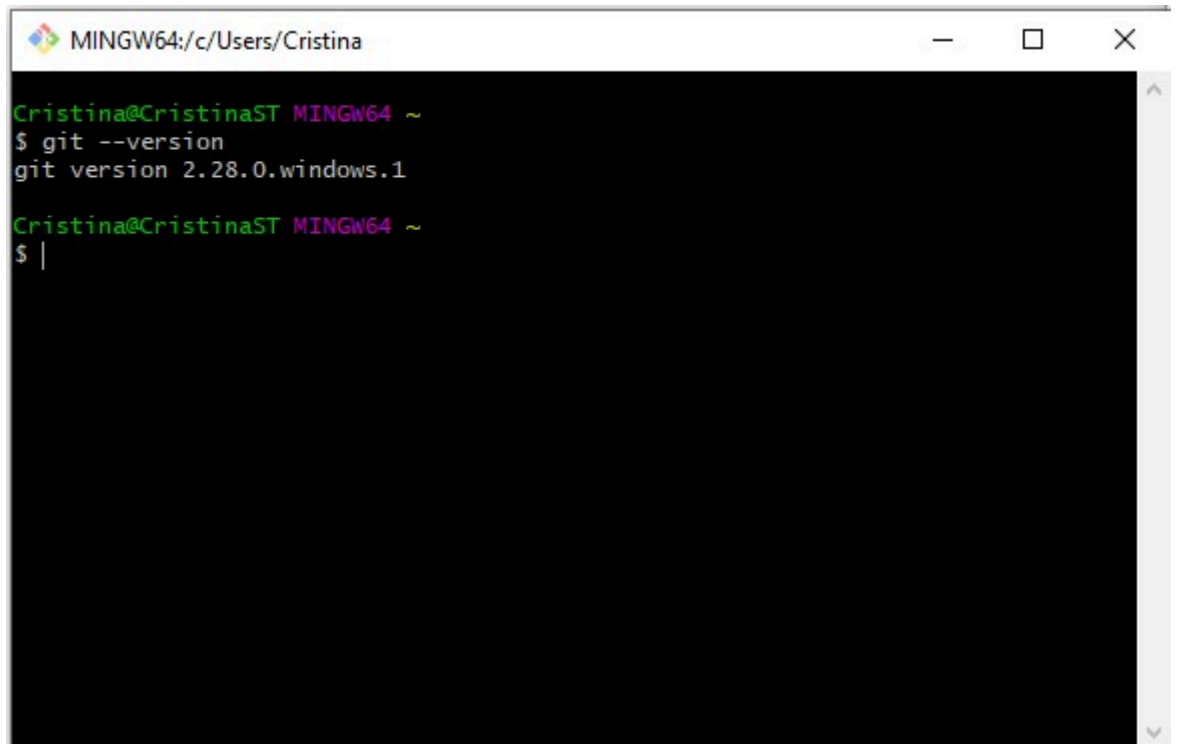


Configurar git

Quan instal·lem GIT entre altres també s'ens instala GIT bash que es una consola.



Aquí també podem veure la versió que tenim de GIT amb la comanda `GIT --version`



```
MINGW64:/c/Users/Cristina

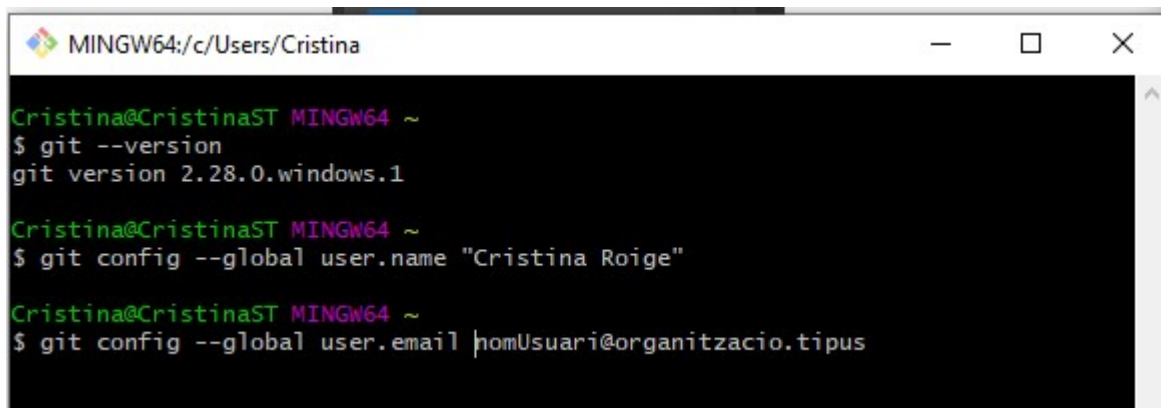
Cristina@CristinaST MINGW64 ~
$ git --version
git version 2.28.0.windows.1

Cristina@CristinaST MINGW64 ~
$ |
```

Configurarem GIT amb el nom d'usuari i el compte de correu electrònic ja que tots els commits de GIT utilitzaran aquesta informació.

```
git config --global user.name "nom usuari"
```

```
git config --global user.email nomUsuari@organitzacio.tipus
```



```
MINGW64:/c/Users/Cristina

Cristina@CristinaST MINGW64 ~
$ git --version
git version 2.28.0.windows.1

Cristina@CristinaST MINGW64 ~
$ git config --global user.name "Cristina Roige"

Cristina@CristinaST MINGW64 ~
$ git config --global user.email nomUsuari@organitzacio.tipus
```

Amb la variable `--global` GIT utilitza informació per tot el sistema. també podem configurar:

- Editor de text → `git config --global core.editor nomEditor`
- Eina per resoldre conflictes per resoldre **merge** → `git config --global merge.tool nomEina`

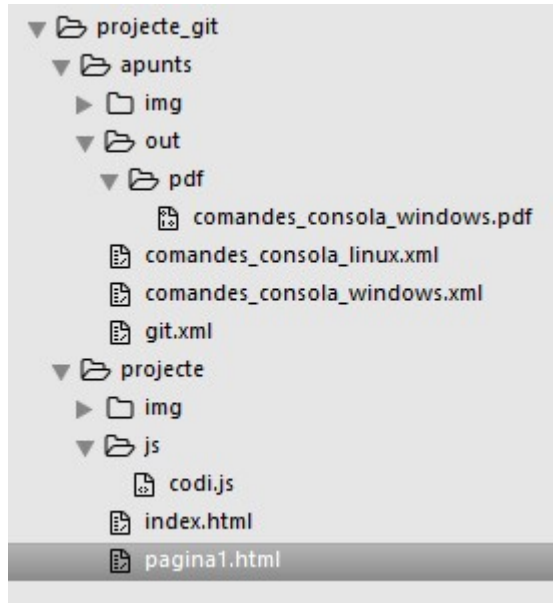
Per comprovar la configuració hem de fer la següent comanda → `git config --list`

Per poder veure les comandes que té GIT → `git help`

Crear repositori

Per crear un repositori el primer que hem fet és crear una carpeta a qualsevol lloc que volguem i dins aquesta hem fet els següents fitxers:

- Carpeta apunts en aquesta carpeta hi hauran els apunts que estic fent per GIT.
- Carpeta projecte on hi haurà un exemple de projecte web



De moment tenim text a index.html i pagina1.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8"/>
  <title>Hola m&oacute;n</title>
  <script type="text/javascript" src="js/codi.js"></script>
</head>
<body>
  <div><h1>Index git</h1></div>
  <div>
    <ul>
      <li><a href="pagina1.html">Pàgina 1</a></li>
    </ul>
  </div>
</body>
</html>
```

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8"/>
  <title>Hola m&ocute;n</title>
  <script type="text/javascript" src="js/codi.js"></script>
</head>
<body>
  <div><h1>Pagina 1</h1></div>
  <div>
    <ul>
      <li><a href="index.html">inici</a></li>
    </ul>
  </div>
</body>
</html>
```

Per iniciar un repositori (projecte) anirem a la carpeta que tinguem el projecte:

```
Cristina@CristinaST MINGW64 ~
$ cd Desktop/projecte_git/
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git
$ ls
apunts/  projecte/
```

Per iniciar un repositori hem d'escriure la comanda de GIT → `git init` podem veure que s'ens ha creat una carpeta oculta que es diu `.git`.

Per llistar els fitxer la comanda és `ls` i si volem veure els fitxers ocults `ls -la`

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git
$ git init
Initialized empty Git repository in C:/Users/Cristina/Desktop/projecte_git/.git/

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ ls
apunts/  projecte/
```

```
.git
apunts
projecte
```

Clonar repositoris

Podem crear un repositori clonant un que ja existeixi i es pot fer de dues formes o clonant un repositori local o bé un repositori remot.

Clonar repositori local → `git clone ./rutaRelativa nomNovaCarpeta`. es clonarà però es clonarà buit.

```
Cristina@CristinaST MINGW64 ~/Desktop
$ git clone ./projecte_git/ projecte_git_clonat
Cloning into 'projecte_git_clonat'...
warning: You appear to have cloned an empty repository.
done.

Cristina@CristinaST MINGW64 ~/Desktop
$ cd projecte_git_clonat/

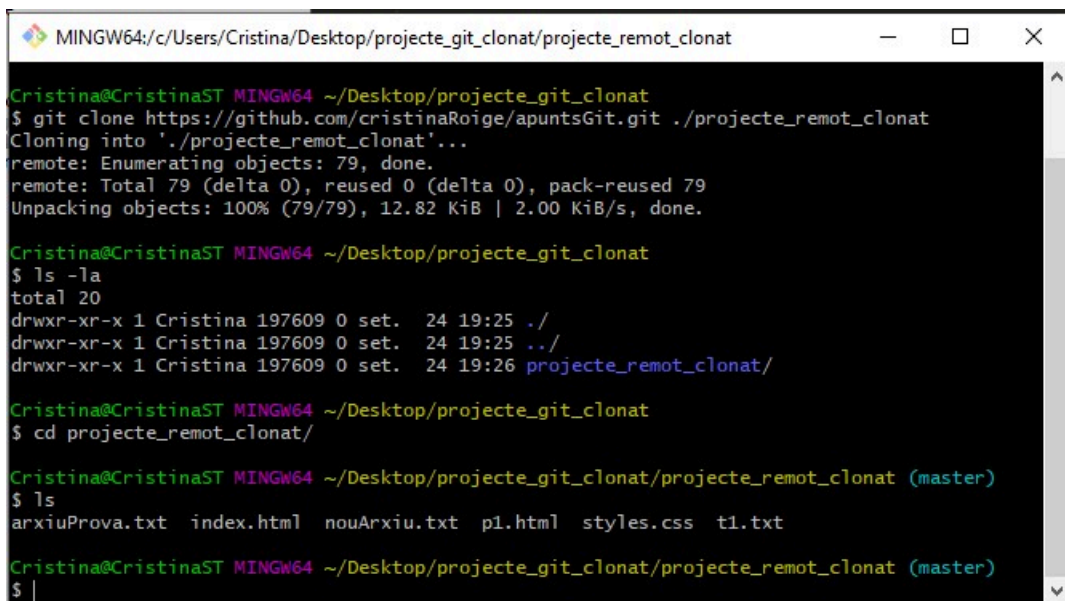
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git_clonat (master)
$ ls

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git_clonat (master)
$ ls -la
total 20
drwxr-xr-x 1 Cristina 197609 0 set. 24 19:01 ./
drwxr-xr-x 1 Cristina 197609 0 set. 24 19:01 ../
drwxr-xr-x 1 Cristina 197609 0 set. 24 19:01 .git/
```

Per clonar un repositori que tinguem a internet:

`git clone https://github.com/nomrepositori/nomProjecte.git ./nomNovaCarpeta`.

Podem veure que el nostre repositori es clonarà.



```
MINGW64:/c/Users/Cristina/Desktop/projecte_git_clonat/projecte_remot_clonat

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git_clonat
$ git clone https://github.com/cristinaRoige/apuntsGit.git ./projecte_remot_clonat
Cloning into './projecte_remot_clonat'...
remote: Enumerating objects: 79, done.
remote: Total 79 (delta 0), reused 0 (delta 0), pack-reused 79
Unpacking objects: 100% (79/79), 12.82 KiB | 2.00 KiB/s, done.

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git_clonat
$ ls -la
total 20
drwxr-xr-x 1 Cristina 197609 0 set. 24 19:25 ./
drwxr-xr-x 1 Cristina 197609 0 set. 24 19:25 ../
drwxr-xr-x 1 Cristina 197609 0 set. 24 19:26 projecte_remot_clonat/

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git_clonat
$ cd projecte_remot_clonat/

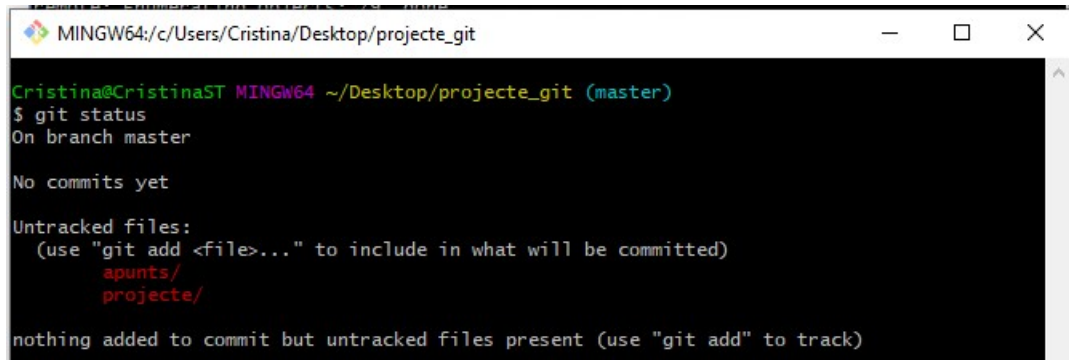
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git_clonat/projecte_remot_clonat (master)
$ ls
arxiuProva.txt index.html nouArxiu.txt p1.html styles.css t1.txt

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git_clonat/projecte_remot_clonat (master)
$ |
```

Espai de treball a GIT

La comanda `git status` mostra tot el que hi ha dins de l'index.

Ens indica a quina branca estem, en aquest cas la branch master ens diu que no hem fet cap commit i els untracked files (fitxers que no estan inclosos per fer el commit, directori de treball).



```

MINGW64; c:/Users/Cristina/Desktop/projecte_git
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git status
On branch master

No commits yet

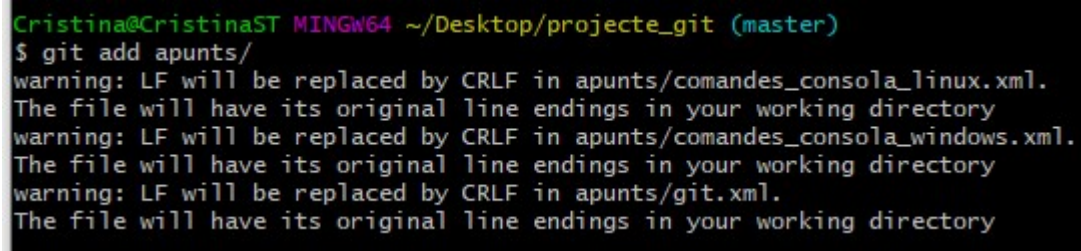
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        apunts/
        projecte/

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

```

Si volem fer un commit amb algun d'aquest fitxers el que hem de fer es fer servir la següent comanda:

```
git add nomCarpetaODirectori
```

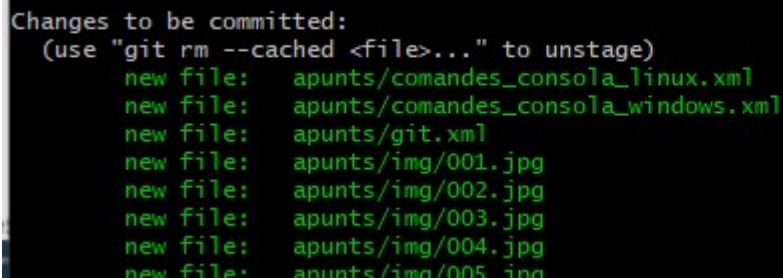


```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git add apunts/
warning: LF will be replaced by CRLF in apunts/comandes_consola_linux.xml.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in apunts/comandes_consola_windows.xml.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in apunts/git.xml.
The file will have its original line endings in your working directory

```

Si ara fem un `git status` podem veure la safata de sortida (stading index) els fitxers i carpetes que estan apunt de rebre el commit.

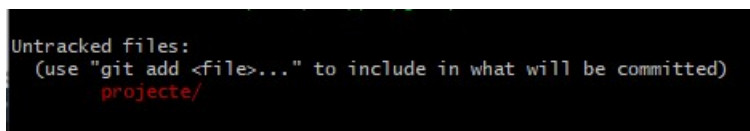


```

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   apunts/comandes_consola_linux.xml
        new file:   apunts/comandes_consola_windows.xml
        new file:   apunts/git.xml
        new file:   apunts/img/001.jpg
        new file:   apunts/img/002.jpg
        new file:   apunts/img/003.jpg
        new file:   apunts/img/004.jpg
        new file:   apunts/img/005.jpg

```

També podem veure untracked files. aquestes carpetes i fitxers estan pendents de passar a la safata de sortida i si ara mateix fessim un commit aquest fitxers no s'inclourien en aquest.



```

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

```

Si volguessim esborrar algun fitxer de la safata de sortida i no incloure'l en el següent commit hauriem de fer servir la comanda:

```
git rm --cached <nomArxiuODirectori>
```

Si volguessim afegir tots els fitxers de cop al següent commit fariem servir la comanda

```
git add .
```

Així que:

- El directori de GIT (GIT directory) → Emmagatzema les metadades i la base de dades d'objectes per el projecte. És la part més important de GIT, i és el que es copia quan es clona un repositori des d'un altre ordinador.

- El directori de treball (working directori)→ És una còpia d'una versió del projecte. Aquests arxius es treuen de la base de dades comprimida en el directori de GIT, I es col·loquen en disc perquè es puguin utilitzar o modificar.
- L'àrea de preparació (standing area): és un senzill arxiu, generalment contingut en el directori de GOT que emmagatzema informació sobre el que va a anar a la propera confirmació. De vegades es denomina index, però s'està convertint en estàndard el referir -s'hi com l'àrea de preparació.

Si una versió concreta d'un arxiu està en el directori de GIT, es considera confirmada (Committed). Si ha sofert canvis des que es va obtenir del repositori, però ha estat afegida a l'àrea de preparació, està preparada (staged). I si ha sofert canvis des que es va obtenir el repositori, però no s'ha preparat, està modificada (modified).

Comandes status add dif

Si fem dos canvis als fitxers index.html i pagina1.html i després fem un GIT status podem veure que aquests dos fitxers estan al nostre directori de treball i no estaran inclosos en el commit. si fem GIT add al fitxer index.html podem veure que només ens queda el fitxer pagina1.html al directori de treball.

```
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:   projecte/index.html
        modified:   projecte/pagina1.html
```

```
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:   projecte/index.html
        modified:   projecte/pagina1.html
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git add projecte/index.html
```

```
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:   projecte/pagina1.html
```

Si no ens agraden els canvis que hem fet en aquesta pàgina podem utilitzar la comanda

```
git checkout pagina1.html
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git checkout projecte/pagina1.html
Updated 1 path from the index
```

podem veure que a la safata de sortida ja no ens hi queda res.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$
```


Si tornem a fer canvis a la pàgina1.html guardem i a la consola tornem a fer GIT status podem veure que la pagina1.html s'ha modificat.

```
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:   projecte/pagina1.html

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
```

Si volem veure els canvis que s'han produït hem d'utilitzar la comanda

```
git diff nomCarpetaOArxiu
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git diff projecte/pagina1.html
diff --git a/projecte/pagina1.html b/projecte/pagina1.html
index 1b6081a..884e682 100644
--- a/projecte/pagina1.html
+++ b/projecte/pagina1.html
@@ -21,6 +21,6 @@
         <li><a href="index.html">inici</a></li>
     </ul>
     </div>
-    <div class="clear"><p>Canvi 1</p></div>
+    <div class="clear"><p>Canvi a la pàgina 1.</p></div>
</body>
</html>
\ No newline at end of file

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
```

En vermell podem veure com estava abans i en verd com està ara.

fitxer gitignore

El fitxer gitignore serveix perquè una carpeta o arxiu no es guardi mai en un commit.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ touch .gitignore

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ nano .gitignore
```

hem d'escriure la ruta de la carpeta des del directori de treball

Si fem `git status` ara podem veure aquest fitxer que podem fer commit i que evita que els fitxers que estan dins no s'afegeixin mai.

Comanda commit i versions

Un commit és una versió del programa actual, un punt de partida al nostre codi que podrem tornar sempre que vulguem i comparar-lo amb altres versions que tinguem.

Un cop tinguem tots els fitxers a la safata de sortida després de fer un add hem d'utilitzar la següent comanda:

```
git commit -m "comentari identificatiu del comit actual"
```

Si fem `git status` veiem que no hi ha cap commit per fer.

Si fem servir la comanda

git log --oneline veurem aquest commit.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline
708b461 (HEAD -> master) commit inicial
```

Si fem git log podrem veure tota la informació del commit

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log
commit 708b4614c7d0fa92bf82ddd0be1c769fb0b1d039 (HEAD -> master)
Author: Cristina Roige <crisinaroige75@gmail.com>
Date: Thu Sep 24 22:23:11 2020 +0200

    commit inicial

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
```

Si fem canvis al fitxer index.html i al fitxer pagina1.html podem *si estem molt segurs que podem fer els canvis* podem fer servir la comanda git commit -am "comentari identificatiu del commit actual.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git status
On branch master
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:   projecte/index.html
        modified:   projecte/pagina1.html

no changes added to commit (use "git add" and/or "git commit -a")

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git commit -am "afegim paragraf index.html i pagina1.html"
[master 40dfa13] afegim paragraf index.html i pagina1.html
2 files changed, 2 insertions(+), 1 deletion(-)

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
```

Si tornem a fer git log podrem veure l'història dels commits.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log
commit 40dfa1304c762bc58b104a04cb87faf5208e9bd7 (HEAD -> master)
Author: Cristina Roige <crisinaroige75@gmail.com>
Date: Thu Sep 24 22:35:34 2020 +0200

    afegim paragraf index.html i pagina1.html

commit 708b4614c7d0fa92bf82ddd0be1c769fb0b1d039
Author: Cristina Roige <crisinaroige75@gmail.com>
Date: Thu Sep 24 22:23:11 2020 +0200

    commit inicial

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
```

Per comparar dos commits es fa servir la comanda git diff idCommit1 idCommit2

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline
40dfa13 (HEAD -> master) afegim paragraf index.html i pagina1.html
708b461 commit inicial

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git diff 40dfa13 708b461
diff --git a/projecte/index.html b/projecte/index.html
index 708b461..40dfa13
--- a/projecte/index.html
+++ b/projecte/index.html
@@ -1,2 +1,3 @@
<h1>Projecte</h1>
+<p>Paragraf afegit</p>
<p>Paragraf inicial</p>
```

Podem veure els canvis que hi ha hagut entre els dos commits. en vermell el que s'ha canviat en verd el que s'ha eliminat.

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git diff 40dfa13 708b461
diff --git a/projecte/index.html b/projecte/index.html
index 824af0f..65ea575 100644
--- a/projecte/index.html
+++ b/projecte/index.html
@@ -22,6 +22,6 @@
         </ul>
     </div>
     <div class="clear"><p>Canvi 1</p></div>
-    <div class="clear"><p>Hem afegit un paràgraf</p></div>
+
</body>
</html>
\ No newline at end of file
diff --git a/projecte/pagina1.html b/projecte/pagina1.html
index 8d3819b..884e682 100644
--- a/projecte/pagina1.html
+++ b/projecte/pagina1.html
@@ -22,6 +22,5 @@
     </ul>
</div>
<div class="clear"><p>Canvi a la pàgina 1.</p></div>
-    <div class="clear"><p>Hem afegit un paràgraf</p></div>
</body>
...skipping...
diff --git a/projecte/index.html b/projecte/index.html
index 824af0f..65ea575 100644
--- a/projecte/index.html
+++ b/projecte/index.html
@@ -22,6 +22,6 @@
         </ul>
     </div>
     <div class="clear"><p>Canvi 1</p></div>
-    <div class="clear"><p>Hem afegit un paràgraf</p></div>
+
</body>
</html>
\ No newline at end of file
diff --git a/projecte/pagina1.html b/projecte/pagina1.html
index 8d3819b..884e682 100644
--- a/projecte/pagina1.html
+++ b/projecte/pagina1.html
@@ -22,6 +22,5 @@
     </ul>
</div>
<div class="clear"><p>Canvi a la pàgina 1.</p></div>
-    <div class="clear"><p>Hem afegit un paràgraf</p></div>
</body>
</html>
\ No newline at end of file
~
~

```

Comandes reset i checkout

Si fem un canvi a un fitxer, per exemple a index.html si per error fem la comanda `git add .` i ens adonem que ens hem equivocat i el volem tornar al directori de treball podem fer servir la comanda `git reset HEAD nomFitxer` o bé també podem fer servir la comanda `git reset .` o bé `git checkout nomFitxer`


```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git status
On branch master
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:   projecte/index.html

no changes added to commit (use "git add" and/or "git commit -a")

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git add .

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   projecte/index.html

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$

```

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git reset HEAD projecte/index.html
Unstaged changes after reset:
M       projecte/index.html

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git status
On branch master
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:   projecte/index.html

no changes added to commit (use "git add" and/or "git commit -a")

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$

```

Desfer un commit

Per veure la comanda `reset` que es fa servir per desfer un commit hem fet diverses modificacions i diversos commits

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline
cc6230e (HEAD -> master) canvi per veure els resets 3
713a3dd canvi per veure els resets 3
9697feb canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial

```

La comanda `git reset --hard HEAD` reseteja el directori de treball a l'últim estat al que estavem treballant.

Si volem desfer el últim commit, en el nostre cas `cc6230e` hem d'escriure la comanda.


```
git reset --hard HEAD~numCommitsQueVolemTornarEnrera
```

Per exemple en el nostre cas `$ git reset --hard HEAD~1` esborrarà l'últim commit.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline
cc6230e (HEAD -> master) canvi per veure els resets 3
713a3dd canvi per veure els resets 3
9697feb canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git reset --hard HEAD~1
HEAD is now at 713a3dd canvi per veure els resets 3

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline
713a3dd (HEAD -> master) canvi per veure els resets 3
9697feb canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial
```

Hem de tindre en compte que els commits es compten a partir de 0.

Si volem esborrar uns commit farem `git reset --hard idComit` això ens esborrarà des del HEAD fins al commit anterior al que hem posat de ID

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline
713a3dd (HEAD -> master) canvi per veure els resets 3
9697feb canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git reset --hard c7f05b5
HEAD is now at c7f05b5 canvi per veure els resets 1

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline
c7f05b5 (HEAD -> master) canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
```

Si no fem servir el parametre hard la diferència és que desfem el commit però el codi queda intacte als fitxers.

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline
b07ba7b (HEAD -> master) canvi per veure els resets 4
31f2096 canvi per veure els resets 3
e4cbdd7 canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git diff b07ba7b 31f2096
diff --git a/projecte/index.html b/projecte/index.html
index ed740f8..db6adca 100644
--- a/projecte/index.html
+++ b/projecte/index.html
@@ -27,6 +27,5 @@
     <div class="clear"><p>Hem afegit el paràgraf 3</p></div>
     <div class="clear"><p>Hem afegit el paràgraf 4</p></div>
     <div class="clear"><p>Hem afegit el paràgraf 5</p></div>
-    <div class="clear"><p>Hem afegit el paràgraf 6</p></div>
   </body>
 </html>
\ No newline at end of file

```

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git reset 31f2096
Unstaged changes after reset:
M   projecte/index.html

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline
31f2096 (HEAD -> master) canvi per veure els resets 3
e4cbdd7 canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$

```

Tornar a un commit anterior

Si volem canviar el HEAD dels commits i que aquest sigui el nostre últim commit hem de fer servir la comanda

```
git checkout idCommit
```

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline
a2d2bfd (HEAD -> master) recuperar commit 3
7659c74 recuperar commit 2
a470d65 recuperar commit 1
53fd94b git.xml tornar commit anterior
31f2096 canvi per veure els resets 3
e4cbdd7 canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git checkout 31f2096
Note: switching to '31f2096'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

    git switch -c <new-branch-name>

Or undo this operation with:

    git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at 31f2096 canvi per veure els resets 3
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git ((31f2096...))

```

Els commits anteriors no han desaparegut i els podem tornar a recuperar.

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git ((31f2096...))
$ git log --oneline
31f2096 (HEAD) canvi per veure els resets 3
e4cbdd7 canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git ((31f2096...))

```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git ((31f2096...))
$ git checkout a2d2bfd
Previous HEAD position was 31f2096 canvi per veure els resets 3
HEAD is now at a2d2bfd recuperar commit 3

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git ((a2d2bfd...))
$ git log --oneline
a2d2bfd (HEAD, master) recuperar commit 3
7659c74 recuperar commit 2
a470d65 recuperar commit 1
53fd94b git.xml tornar commit anterior
31f2096 canvi per veure els resets 3
e4cbdd7 canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git ((a2d2bfd...))
$ git status
HEAD detached at a2d2bfd
nothing to commit, working tree clean

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git ((a2d2bfd...))
```

Buscar un commit

Per buscar un commit en concret

```
git log --grep="paraules clau"
```

```
git log --oneline --grep="paraules clau"
```



```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline
854013f (HEAD -> master) git.xml buscar un commit
a2d2bfd recuperar commit 3
7659c74 recuperar commit 2
a470d65 recuperar commit 1
53fd94b git.xml tornar commit anterior
31f2096 canvi per veure els resets 3
e4cbdd7 canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --grep="veure els"
commit 31f20962e61b18ad292bf5481eb6aee0b7e246c5
Author: Cristina Roige <crisinaroige75@gmail.com>
Date: Thu Sep 24 23:50:42 2020 +0200

    canvi per veure els resets 3

commit e4cbdd7886231de9c727ac28d65a9c11e543e14e
Author: Cristina Roige <crisinaroige75@gmail.com>
Date: Thu Sep 24 23:50:09 2020 +0200

    canvi per veure els resets 2

commit c7f05b526cdda7ab010fb40a2b04341a6afb7679
Author: Cristina Roige <crisinaroige75@gmail.com>
Date: Thu Sep 24 23:21:49 2020 +0200

    canvi per veure els resets 1

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline --grep="veure els"
31f2096 canvi per veure els resets 3
e4cbdd7 canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
```

També podem buscar els commits per el contingut dels fitxers.

```
git log -S "paraules clau"
```

```
git log --oneline -S "paraules clau"
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log -S "Hem afegit el paràgraf 4"
commit a2d2bfd54148ed0a9e6a6baa44c8aa12fa6891f0
Author: Cristina Roige <cristinaroige75@gmail.com>
Date:   Fri Sep 25 00:08:16 2020 +0200

    recuperar commit 3

commit e4cbdd7886231de9c727ac28d65a9c11e543e14e
Author: Cristina Roige <cristinaroige75@gmail.com>
Date:   Thu Sep 24 23:50:09 2020 +0200

    canvi per veure els resets 2

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline -S "Hem afegit el paràgraf 4"
a2d2bfd recuperar commit 3
e4cbdd7 canvi per veure els resets 2

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
```

Branch

Les branques (branch) de desenvolupament permeten ramificar el nostre projecte i tindre diferents línies de desenvolupament, podent desenvolupar altres funcionalitats sense que unes interfereixin amb les altres.

El més normal és tindre una branch master i mentres anem desenvolupant el projecte sorgeixen noves funcionalitats i nosaltres no volem barrejar aquestes funcionalitats amb la branch master o qualsevol altre branca de desenvolupament.

Per crear una nova branca farem servir:

```
git branch nomBranca
```

Per veure si s'ha creat la branca:

```
git branch
```

Per canviar d'una branca a una altra branca

```
git checkout nomBranca
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
($ git branch pagina1

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git branch
* master
  pagina1

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git checkout pagina1
Switched to branch 'pagina1'

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git status
On branch pagina1
nothing to commit, working tree clean

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
```


Fem els canvis preparem i fem el commit i quan fem git log --oneline podem veure que estem a la branca que hem creat que allí hem fet el nostre commit. i la branca master i la branca pagina1 no es el mateix.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git status
On branch pagina1
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:   projecte/pagina1.html

no changes added to commit (use "git add" and/or "git commit -a")

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git add .

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git status
On branch pagina1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   projecte/pagina1.html

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git commit -m "canvi a la pagina1"
[pagina1 7ca8c7c] canvi a la pagina1
1 file changed, 1 insertion(+)

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git log --oneline
7ca8c7c (HEAD -> pagina1) canvi a la pagina1
854013f (master) git.xml buscar un commit
a2d2bfd recuperar commit 3
7659c74 recuperar commit 2
a470d65 recuperar commit 1
53fd94b git.xml tornar commit anterior
31f2096 canvi per veure els resets 3
e4cbdd7 canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial
```

Crearem una branca nova per veure la comanda d'esborrar. aquesta branca es dirà brancaError

```
git branch -D nomBrancaAEsborrar
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git branch
brancaError
master
* pagina1

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git branch -D brancaError
Deleted branch brancaError (was 7ca8c7c).

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git branch
master
* pagina1
```

Si hi hagués algun commit fet dins la branca no ens ho deixaria esborrar ni canviar de branca i primer ens faria un commit.

Quan hagim acabat de treballar a una branca i volguem passar el nostre codi a la branca master haurem d'anar a la nostra branca principal i fer un merge, el merge també serveix per passar el codi de la branca master a la branca que esteu treballant.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git checkout master
Switched to branch 'master'

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git status
On branch master
nothing to commit, working tree clean
```

`git merge nomBranca`

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git merge pagina1
Updating 854013f..7ca8c7c
Fast-forward
 projecte/pagina1.html | 1 +
 1 file changed, 1 insertion(+)
```

Si ara faig un git log veurem que haurem importat els commits fets a la branca que hem importat a la branca master.

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline
854013f (HEAD -> master) git.xml buscar un commit
a2d2bfd recuperar commit 3
7659c74 recuperar commit 2
a470d65 recuperar commit 1
53fd94b git.xml tornar commit anterior
31f2096 canvi per veure els resets 3
e4cbdd7 canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git merge pagina1
Updating 854013f..7ca8c7c
Fast-forward
 projecte/pagina1.html | 1 +
 1 file changed, 1 insertion(+)

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline
7ca8c7c (HEAD -> master, pagina1) canvi a la pagina1
854013f git.xml buscar un commit
a2d2bfd recuperar commit 3
7659c74 recuperar commit 2
a470d65 recuperar commit 1
53fd94b git.xml tornar commit anterior
31f2096 canvi per veure els resets 3
e4cbdd7 canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial

```

Comanda stash

la comanda stash permet canviar-te a una altra branca que estiguis treballant sense emportar-te els canvis que has fet a la pàgina que estaves treballant.

Imaginem-nos que estem treballant a la branca principal al fitxer index.html i li fem un canvi. ens ve el jefe i diu deixa el que estas fent i arregla'm la pàgina1.html. el que hauriem de fer és un stash que ens guardaria els canvis fets al fitxer index.html i els podrem guardar més endavant.

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git stash
Saved working directory and index state WIP on master: f3d7f98 git.xml git stash

```

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git status
On branch master
nothing to commit, working tree clean

```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git stash show
projecte/index.html | 1 +
1 file changed, 1 insertion(+)
```

Ens podem canviar de branca i si fem un `git status` podem veure que no ens hem emportat res de l'altra branca.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git stash show
projecte/index.html | 1 +
1 file changed, 1 insertion(+)
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git branch
* master
  pagina1
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git checkout pagina1
Switched to branch 'pagina1'
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git status
On branch pagina1
nothing to commit, working tree clean
```

La memoria stash es global així que estant a la branca `pagina1` fem `git stash show` veurem que els canvis continuen pendents

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git stash show
projecte/index.html | 1 +
1 file changed, 1 insertion(+)
```

Fem els canvis que ens han demanat.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git status
On branch pagina1
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:   projecte/pagina1.html

no changes added to commit (use "git add" and/or "git commit -a")
```

Podem fer `add` i `commit`


```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git add .

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git status
On branch pagina1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   projecte/pagina1.html

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git commit -m "canvis ultima hora"
[pagina1 c2bb097] canvis ultima hora
1 file changed, 1 insertion(+)

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git status
On branch pagina1
nothing to commit, working tree clean

```

Podem veure que tenim el commit fet

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (pagina1)
$ git log --oneline
c2bb097 (HEAD -> pagina1) canvis ultima hora
7ca8c7c canvi a la pagina1
854013f git.xml buscar un commit
a2d2bfd recuperar commit 3
7659c74 recuperar commit 2
a470d65 recuperar commit 1
53fd94b git.xml tornar commit anterior
31f2096 canvi per veure els resets 3
e4cbdd7 canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial

```

Tornem a la branca master i podem veure que en el stash encara tenim els canvis pendents

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git stash show
projecte/index.html | 1 +
1 file changed, 1 insertion(+)

```

Treiem el que teniem a la memoria i els recuperem a l'index

git stash pop

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git stash pop
On branch master
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:   projecte/index.html

no changes added to commit (use "git add" and/or "git commit -a")
Dropped refs/stash@{0} (ddd7f960b8aea55b6bd0cc623ab30ba2fbf72f91)

```

Per netejar la memoria stash `git stash clear`

Comanda cherry-pick

La comanda cherry-pick serveix per fer un merge parcial, això significa que podem estar treballant en una branca, fer la nostra feina amb els commits que ens convinguin i en un moment donat tindre que tornar a la branca que sigui i tindre que exportar només alguns canvis dels que s'estaven fent.

A continuació farem un exemple de la comanda cherry-pick

1. Mirem a la branca que estem i com ho tenim.

Figure 1. comanda → git status.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git status
On branch master
nothing to commit, working tree clean
```

2. Crearem un altre full que anomenarem contacte.html i a la pàgina index.html crearem un link per anar aquesta fulla i tornarem a mirar com tenim el nostre directori de treball (working directory) i el nostre index (stage)

Figure 2. comanda → git status.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git status
On branch master
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:   projecte/index.html

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

no changes added to commit (use "git add" and/or "git commit -a")
```

3. Preparem els dos arxius per el commit

Figure 3. comanda → git add nomArxiu nomArxiu.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git add projecte/index.html projecte/contacte.html
```

4. Fem el commit

Figure 4. comanda → git commit -m "missatge descriptiu commit".

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git commit -m "Creació contacte.html i link index.html"
[master b32007e] Creació contacte.html i link index.html
2 files changed, 26 insertions(+)
create mode 100644 projecte/contacte.html
```


5. Crearem una nova branca per crear un formulari de contacte

Figure 5. comanda → `git add nomArxiu nomArxiu.`

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git branch contacte
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
```

6. Mirarem com tenim el directori de treball i les branch

Figure 6. comanda → `git status.i git branch`

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git status
On branch master
nothing to commit, working tree clean

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git branch
contacte
* master
paginal

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
```

7. Ens canviarem de branca

Figure 7. comanda → `git checkout nomBrancaACanviar.i git status`

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git checkout contacte
Switched to branch 'contacte'

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (contacte)
$ git status
On branch contacte
nothing to commit, working tree clean

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (contacte)
$ ls -la
total 33
drwxr-xr-x 1 Cristina 197609  0 set.  24 22:03 ./
drwxr-xr-x 1 Cristina 197609  0 set.  24 20:10 ../
drwxr-xr-x 1 Cristina 197609  0 set.  25 10:33 .git/
-rw-r--r-- 1 Cristina 197609 25 set.  25 10:13 .gitignore
drwxr-xr-x 1 Cristina 197609  0 set.  25 09:35 apunts/
drwxr-xr-x 1 Cristina 197609  0 set.  25 10:04 projecte/
```

8. Afegim un quants commits des d'aquesta branca comprovem el directori de treball i els commits que hem fet. veiem que hem fet un formulari amb nom i contrasenya i un submit

Figure 8. comanda → `git status.git log --oneline`

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (contacte)
$ git status
On branch contacte
nothing to commit, working tree clean

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (contacte)
$ git log --oneline
9313819 (HEAD -> contacte) Afegim 3 paràgraf
818c1b6 Input submit form contacte
d862e90 Input Contrasenya form contacte
834af17 Input Nom form contacte
b32007e (master) Creació contacte.html i link index.html
5853cc5 modificant .gitignore i git.xml
b674d06 actualitzacio apunts i projecte
f3d7f98 git.xml git stash
7ca8c7c canvi a la pagina1
854013f git.xml buscar un commit
a2d2bfd recuperar commit 3
7659c74 recuperar commit 2
a470d65 recuperar commit 1
53fd94b git.xml tornar commit anterior
31f2096 canvi per veure els resets 3
e4cbdd7 canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i pagina1.html
708b461 commit inicial

```

9. Ens canviarem de branca

Figure 9. comanda → `git checkout nomBrancaACanviar.i git status`

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (contacte)
$ git checkout master
Switched to branch 'master'

```

10. Si mirem els commits podem comparar les dues branques, podem veure que a la branca master no hi ha els commits que hem fet a la branca contacte.

Figure 10. comanda → `git log --oneline`

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (contacte)
$ git log --oneline
9313819 (HEAD -> contacte) Afegim 3 paràgraf
818c1b6 Input submit form contacte
d862e90 Input Contrasenya form contacte
834af17 Input Nom form contacte
b32007e (master) Creació contacte.html i link index.html
5853cc5 modificant .gitignore i git.xml
b674d06 actualitzacio apunts i projecte
f3d7f98 git.xml git stash
7ca8c7c canvi a la paginal
854013f git.xml buscar un commit
a2d2bfd recuperar commit 3
7659c74 recuperar commit 2
a470d65 recuperar commit 1
53fd94b git.xml tornar commit anterior
31f2096 canvi per veure els resets 3
e4cbdd7 canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i paginal.html
708b461 commit inicial

```

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git log --oneline
b32007e (HEAD -> master) Creació contacte.html i link index.html
5853cc5 modificant .gitignore i git.xml
b674d06 actualitzacio apunts i projecte
f3d7f98 git.xml git stash
7ca8c7c canvi a la paginal
854013f git.xml buscar un commit
a2d2bfd recuperar commit 3
7659c74 recuperar commit 2
a470d65 recuperar commit 1
53fd94b git.xml tornar commit anterior
31f2096 canvi per veure els resets 3
e4cbdd7 canvi per veure els resets 2
c7f05b5 canvi per veure els resets 1
067bf36 git.xml section 10
10d1bd4 actualitzacio fitxer git.xml
40dfa13 afegim paragraf index.html i paginal.html
708b461 commit inicial

```

11. Volem que els commits referents al formulari passin a la branca master, però que els que afegeixen els 3 paràgraf no.

Figure 11. comanda → `git cherry-pick idCommit`

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git cherry-pick 9313819
Auto-merging projecte/contacte.html
CONFLICT (content): Merge conflict in projecte/contacte.html
error: could not apply 9313819... Afegim 3 paràgraf
hint: after resolving the conflicts, mark the corrected paths
hint: with 'git add <paths>' or 'git rm <paths>'
hint: and commit the result with 'git commit'

```

12. Veurem que hi ha un conflicte i l'haurem de resoldre (com resoldre conflictes s'explica després) el resoltem, (només es resol a la branca master).

Figure 12. comanda → `git status`

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master|CHERRY-PICKING)
$ git status
On branch master
You are currently cherry-picking commit 9313819.
(fix conflicts and run "git cherry-pick --continue")
(use "git cherry-pick --skip" to skip this patch)
(use "git cherry-pick --abort" to cancel the cherry-pick operation)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
    both modified:   projecte/contacte.html

no changes added to commit (use "git add" and/or "git commit -a")

```

13. Un cop resolt el conflicte ja podem fer el commit

Figure 13. comanda → `git add .` I `git commit -m "missatge descriptiu commit"`

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master|CHERRY-PICKING)
$ git status
On branch master
You are currently cherry-picking commit 9313819.
  (fix conflicts and run "git cherry-pick --continue")
  (use "git cherry-pick --skip" to skip this patch)
  (use "git cherry-pick --abort" to cancel the cherry-pick operation)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
    both modified:   projecte/contacte.html

no changes added to commit (use "git add" and/or "git commit -a")
```

Resolució de conflictes.

Quan treballem amb GIT I treballem amb diverses branques sobint ens trobarem amb conflictes. Per exemple, estem a la branca `contacte` I afegim un paràgraf.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git checkout contacte
Switched to branch 'contacte'
```

```
<div class="clear"><p>Hem afegit un paràgraf</p></div>
<div class="clear"><p>Hem afegit el paràgraf 3 conflicte a contacte</p></div>
</body>
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (contacte)
$ git commit -m "commit per veure conflicte branch contacte"
[contacte f048bdd] commit per veure conflicte branch contacte
1 file changed, 1 insertion(+), 1 deletion(-)
```

Canviarem a la branca `master`

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (contacte)
$ git checkout master
Switched to branch 'master'
```

Canviarem el mateix paràgraf per crear un conflicte.

```
<div class="clear"><p>Canvi a la pagina 1.</p></div>
<div class="clear"><p>Hem afegit un paràgraf</p></div>
<div class="clear"><p>Hem afegit el paràgraf 3 conflicte a master</p></div>
</body>
```

També farem un commit.


```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git add .

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   projecte/contacte.html

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git commit -m "commit per veure conflicte branch master"
[master 7b272cb] commit per veure conflicte branch master
1 file changed, 1 insertion(+), 1 deletion(-)

```

Ara quan fem un merge sorgiria un conflicte.

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git merge contacte
Auto-merging projecte/contacte.html
CONFLICT (content): Merge conflict in projecte/contacte.html
Automatic merge failed; fix conflicts and then commit the result.

```

Per solucionar el conflicte el que haurem de fer és anar al fitxer amb conflicte i solucionar-ho. com podem veure GIT ens ha modificat el fitxer mostrant-nos les dues línies; la de la branch HEAD i la de la branch contacte. per solucionar-ho el que hauriem de fer es modificar el full i triar el que volguem quedar-nos com a correcte

```

<div class="clear"><p>Hem afegit un paràgraf</p></div>
<<<<<< HEAD
    <div class="clear"><p>Hem afegit el paràgraf 3 conflicte a master</p></div>
=====
    <div class="clear"><p>Hem afegit el paràgraf 3 conflicte a contacte</p></div>
>>>>>> contacte
</body>

```

```

<div class="clear"><p>Hem afegit un paràgraf</p></div>
<div class="clear"><p>Hem afegit el paràgraf 3 conflicte resolt</p></div>

```

Si ara fem `git status` veurem que ho tenim al directori de treball amb both modified i que espera rebre un `git add .` i un `commit`

```

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master|MERGING)
$ git status
On branch master
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both modified:   projecte/contacte.html

no changes added to commit (use "git add" and/or "git commit -a")

```

Un cop solucionat el conflicte veurem que un cop fet `git add .` passa a modified i podem fer el merge correctament.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master|MERGING)
$ git add .

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master|MERGING)
$ git status
On branch master
All conflicts fixed but you are still merging.
  (use "git commit" to conclude merge)

Changes to be committed:
  modified:   projecte/contacte.html

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master|MERGING)
$ git commit -m "[MERGE CONFLICT SOLVED]merge solucionat"
[master defae67] [MERGE CONFLICT SOLVED]merge solucionat


Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ |
```

Repositoris remots I GitHub

Per veure els Repositoris remots farem servir GitHub

Crear repositori


Per crear un nou repositori anirem al nostre compte de GitHub I ens loguejarem, Un repositori es on tindrem el nostre projecte I anirem afegint els canvis.

Per crear un nou repositori clicarem al botó new . Escriurem el nom del nostre repositori marcarem que sigui public (Private es pagant) I clicarem el botó Create repository

Create repository


Create a new repository


A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner *  cristinaRoige ▾ / Repository name * ✓

Great repository names are short and unique. projecte_git is available. Need inspiration? How about **bookish-robot**?

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☐ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**
Choose which files not to track from a list of templates. [Learn more.](#)

☐ **Choose a license**
A license tells others what they can and can't do with your code. [Learn more.](#)

[Create repository](#)

Un cop creat el repository hem de seguir les instruccions podem crear un nou repositori o fer un push a un repositori que nosaltres tinguem hi han varies opcions. Farem les instruccions que tenim a la imatge.

...or create a new repository on the command line

```
echo "# projecte_git" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M master
git remote add origin https://github.com/cristinaRoige/projecte_git.git
git push -u origin master
```

Anirem a la consola de comandes I crearem el fitxer README.md aquest fitxer la sintàxi d'aquest fitxer és `echo "TextQueVulguisAlFitxer" >> README.md.`

La comanda `git init` no cal fer-la ja que realment nosaltres ja ho hem fet però si no ho hauriem de fer.

El següent pas serà fer el commit del fitxer README.md

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ echo "# Aquí va el text que vulguis al fitxer README.md" >> README.md

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ cat README.md
# Aquí va el text que vulguis al fitxer README.md
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        README.md

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

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git add .
warning: LF will be replaced by CRLF in README.md.
The file will have its original line endings in your working directory

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   README.md

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git commit -m "commit fitxer README.md"
[master 12a8d9b] commit fitxer README.md
1 file changed, 1 insertion(+)
create mode 100644 README.md
```

Afegirem un origen remot, quan afegim un origen remot el que fem es vincular el nostre repositori local a un servidor remot

```
git remote add origin llocOnTinguemElNostreRepositoriRemot
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git remote add origin https://github.com/cristinaRoige/projecte_git.git
```

Per pujar els canvis al repositori farem servir la comanda **push** que el que farà serà pujar els nostres canvis al servidor remot de GIT o la comanda **pull** que serveix per descarregar els canvis que hi ha al repositori remot al nostre repositori local.

És interessant fer sempre el push des de la nostra branca master.

```
git push origin master
```

```
Username for 'http://github.com':
```

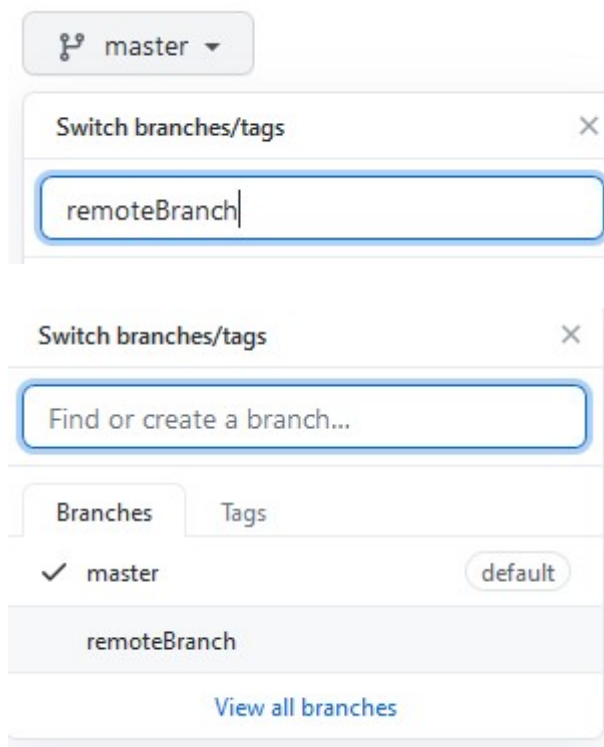
```
Password for 'http://repositoriGitHub@github.com'
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git push origin master
Enumerating objects: 288, done.
Counting objects: 100% (288/288), done.
Delta compression using up to 8 threads
Compressing objects: 100% (276/276), done.
Writing objects: 100% (288/288), 12.33 MiB | 51.00 KiB/s, done.
Total 288 (delta 83), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (83/83), done.
To https://github.com/cristinaRoige/projecte_git.git
 * [new branch]      master -> master
```

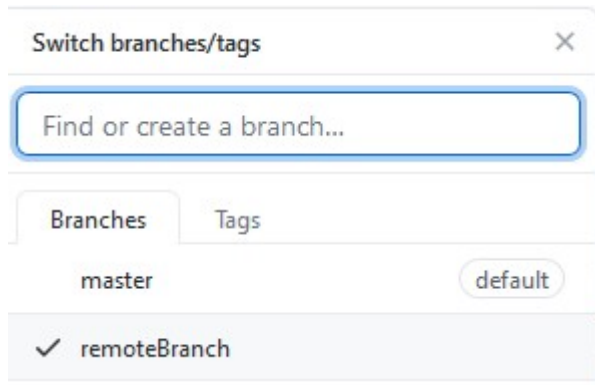
Això treballarà i un cop acabi ja podrem anar al nostre repositori remot que podrem veure el projecte que teniem en local pujat.

	cristinaRoige commit fixer README.md ...	23 minutes ago	🕒 32
📁	apunts	actualitzant repositori	36 minutes ago
📁	projecte	[MERGE CONFLICT SOLVED]merge solucionat	3 hours ago
📄	.gitignore	actualització .gitignore	1 hour ago
📄	README.md	commit fixer README.md	23 minutes ago

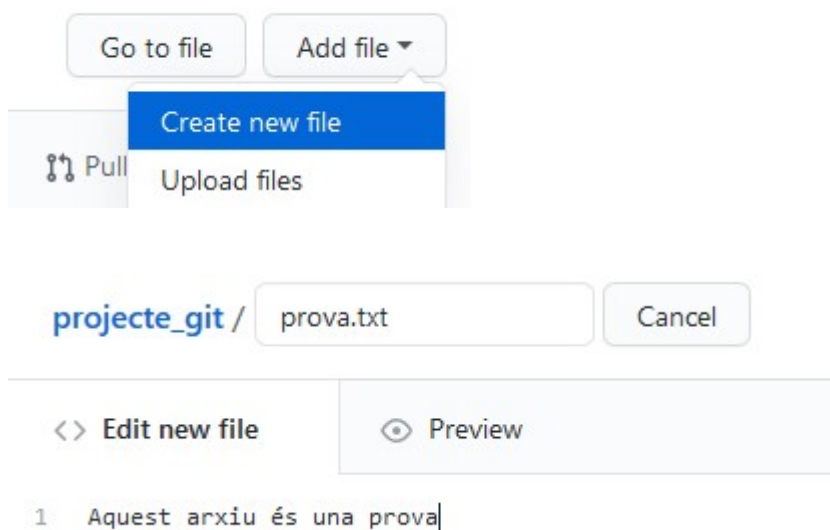
Si volem crear una nova branca des del repositori remot



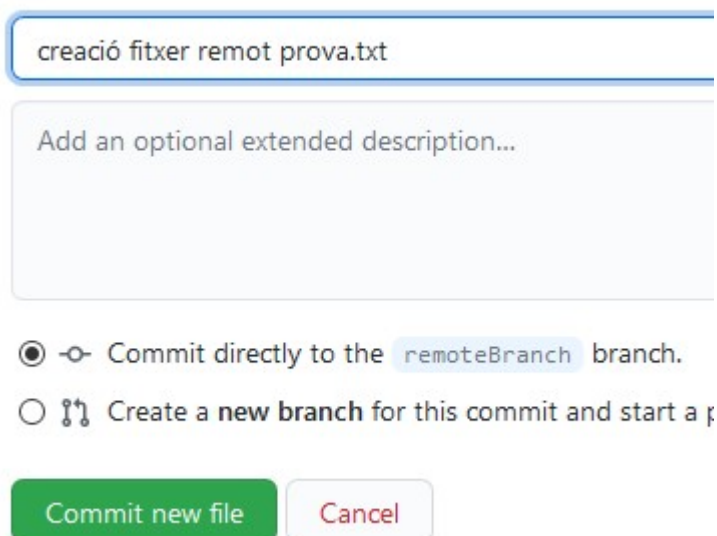
Podem anar a la nova branca clicant sobre aquesta.









Podem afegir un arxiu



Commit new file



El veurem al nostre repositori.

	cristinaRoige creació fitxer remot prova.txt ...	now 33
	apunts actualitzant repositori	1 hour ago
	projecte [MERGE CONFLICT SOLVED]merge solucionat	3 hours ago
	.gitignore actualització .gitignore	1 hour ago
	README.md commit fitxer README.md	1 hour ago
	prova.txt creació fitxer remot prova.txt	now

Podem baixar-nos les branques i fitxers que tenim en remot al nostre repositori local amb la següent comanda fetch:

```
git fetch --all
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git fetch --all
Fetching origin
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 688 bytes | 2.00 KiB/s, done.
From https://github.com/cristinaRoige/projecte_git
* [new branch]      remoteBranch -> origin/remoteBranch
```

Ens podem canviar a la branca que hem baixat del fitxer remot al local i veure el fitxer que hi ha dins.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git checkout remoteBranch
Switched to a new branch 'remoteBranch'
Branch 'remoteBranch' set up to track remote branch 'remoteBranch' from 'origin'
.

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (remoteBranch)
$ git branch
  contacte
  master
  paginal
* remoteBranch

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (remoteBranch)
$ ls -la
total 35
drwxr-xr-x 1 Cristina 197609  0 set.  25 16:59 ./
drwxr-xr-x 1 Cristina 197609  0 set.  24 20:10 ../
drwxr-xr-x 1 Cristina 197609  0 set.  25 16:59 .git/
-rw-r--r-- 1 Cristina 197609 24 set.  25 15:22 .gitignore
drwxr-xr-x 1 Cristina 197609  0 set.  25 15:20 apunts/
drwxr-xr-x 1 Cristina 197609  0 set.  25 13:17 projecte/
-rw-r--r-- 1 Cristina 197609 28 set.  25 16:59 prova.txt
-rw-r--r-- 1 Cristina 197609 50 set.  25 15:45 README.md

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (remoteBranch)
$
```

Podem fer un merge de les dues branques.


```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git merge remoteBranch
Updating 12a8d9b..9f87385
Fast-forward
 prova.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 prova.txt
```

Carregar-nos la branca que hem fet.


```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git branch -D remoteBranch
Deleted branch remoteBranch (was 9f87385).
```

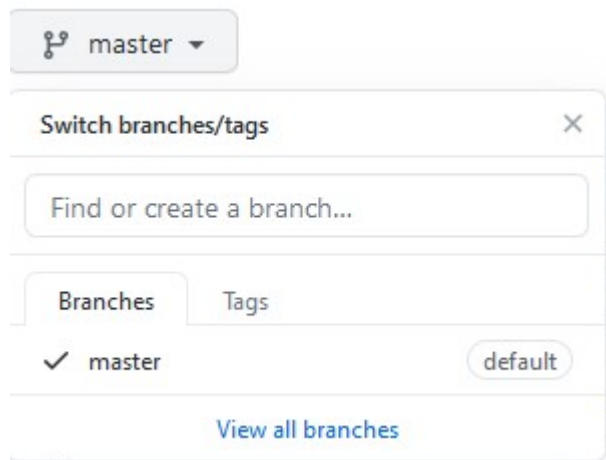
Tornar a fer un push.

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git status
On branch master
nothing to commit, working tree clean

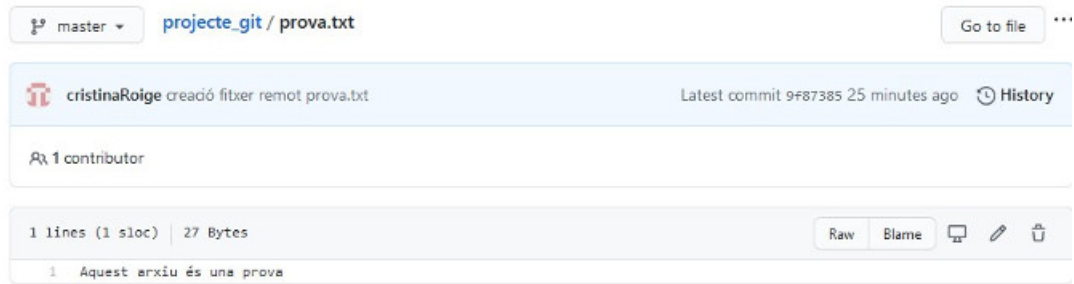
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git push origin master
Enumerating objects: 23, done.
Counting objects: 100% (23/23), done.
Delta compression using up to 8 threads
Compressing objects: 100% (20/20), done.
Writing objects: 100% (20/20), 357.94 KiB | 19.88 MiB/s, done.
Total 20 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/cristinaRoige/projecte_git.git
 12a8d9b..ad6dd47 master -> master
```

podem veure que la branca remoteBranch l'hem eliminat a local però no a remot. si la volem esborrar

per remot clicarem sobre la icona de la paperera. 



Podem fer modificacions en un fitxer en el nostre repositori remot clicant al fitxer que volguem modificar i al llapis i fent el commit..



Un cop tinguem els canvis en aquest fitxer podem fer un pull per baixar aquests canvis al fitxer local.

```
git pull origin master
```

```
Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ git pull origin master
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 709 bytes | 3.00 KiB/s, done.
From https://github.com/cristinaRoige/projecte_git
* branch      master      -> FETCH_HEAD
   ad6dd47..0b4c27f master  -> origin/master
Updating ad6dd47..0b4c27f
Fast-forward
 prova.txt | 4 +++-
 1 file changed, 3 insertions(+), 1 deletion(-)

Cristina@CristinaST MINGW64 ~/Desktop/projecte_git (master)
$ cat prova.txt
Aquest arxiu és una prova
Modificació en remot d'aquest fitxer.
```

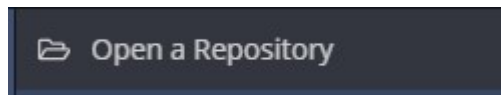
Una eina de gestió gràfica de GIT: gitkraken

Si volem fer servir una eina GUI per gestionar GIT que podem descarregar aquí <https://www.gitkraken.com/> I que ofereix una interfície més amigable per GIT amb integracions a GitHub, gitlab, gitbucket I VSTS(Azure DevOps), I poder interactuar més fàcilment amb el compte de GIT sense utilitzar la línia de comandes.

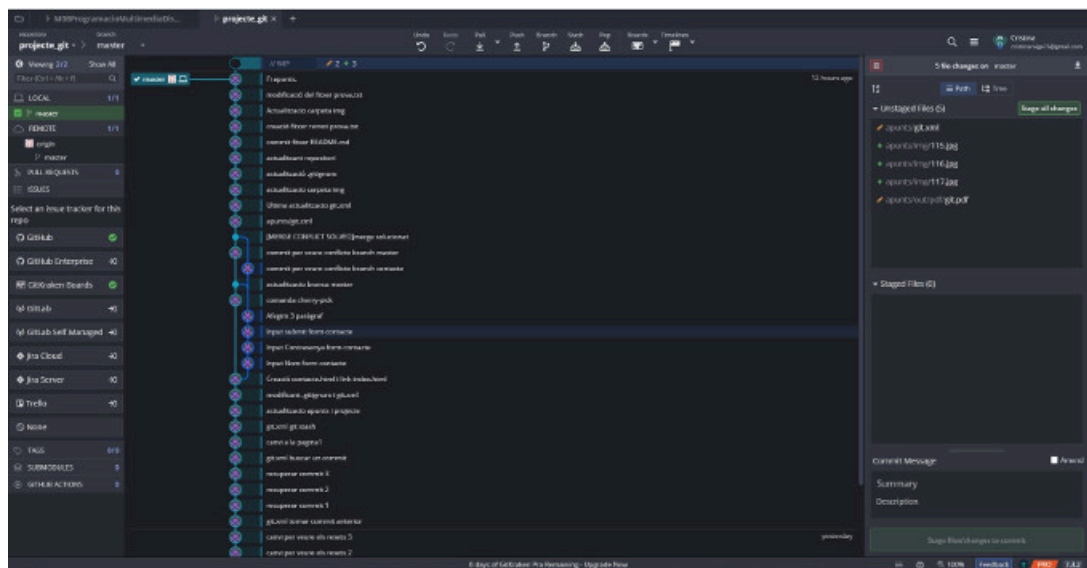
Un cop tens instal·lat gitkraken podem Obrir, clonar o començar un repositori:



En aquest cas per veure breument el que farem és obrir el repositori on tenim els nostres apunts de git. clican a Open a Repositori



Un cop obert podem veure la següent vista:

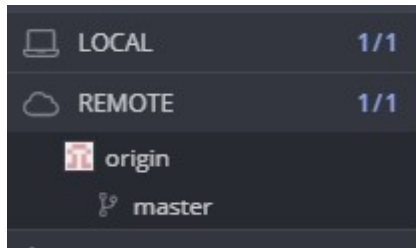


Podem veure les següents parts:

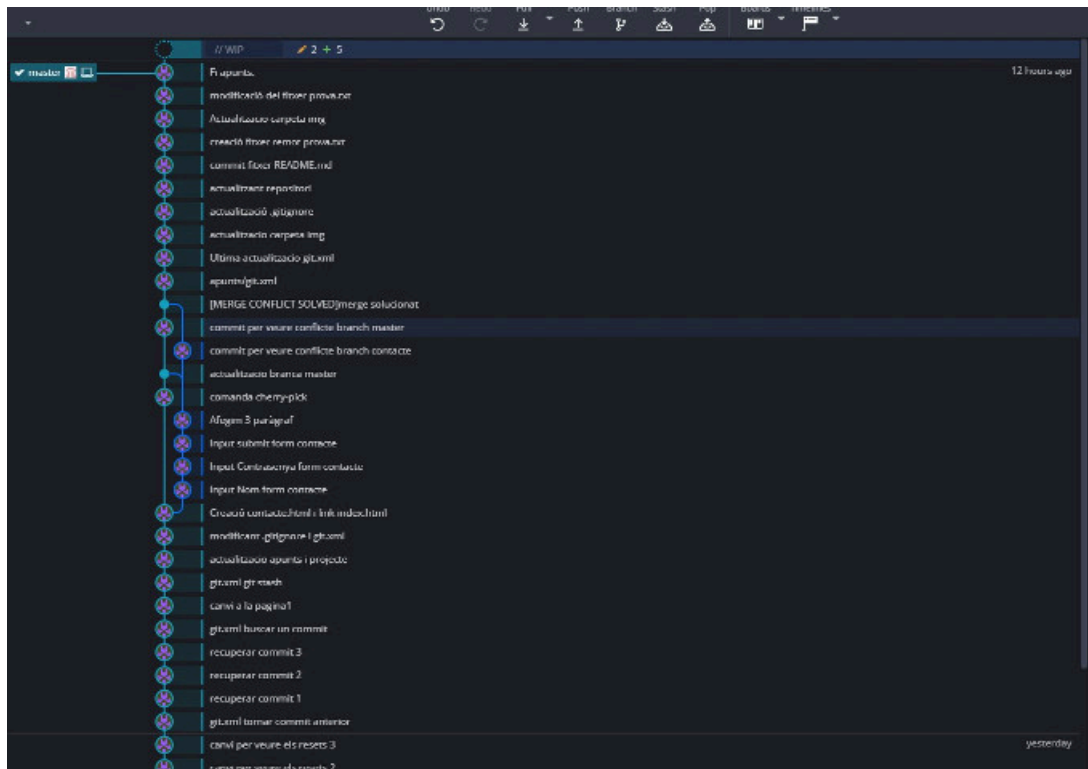
- Branca on estem actualment



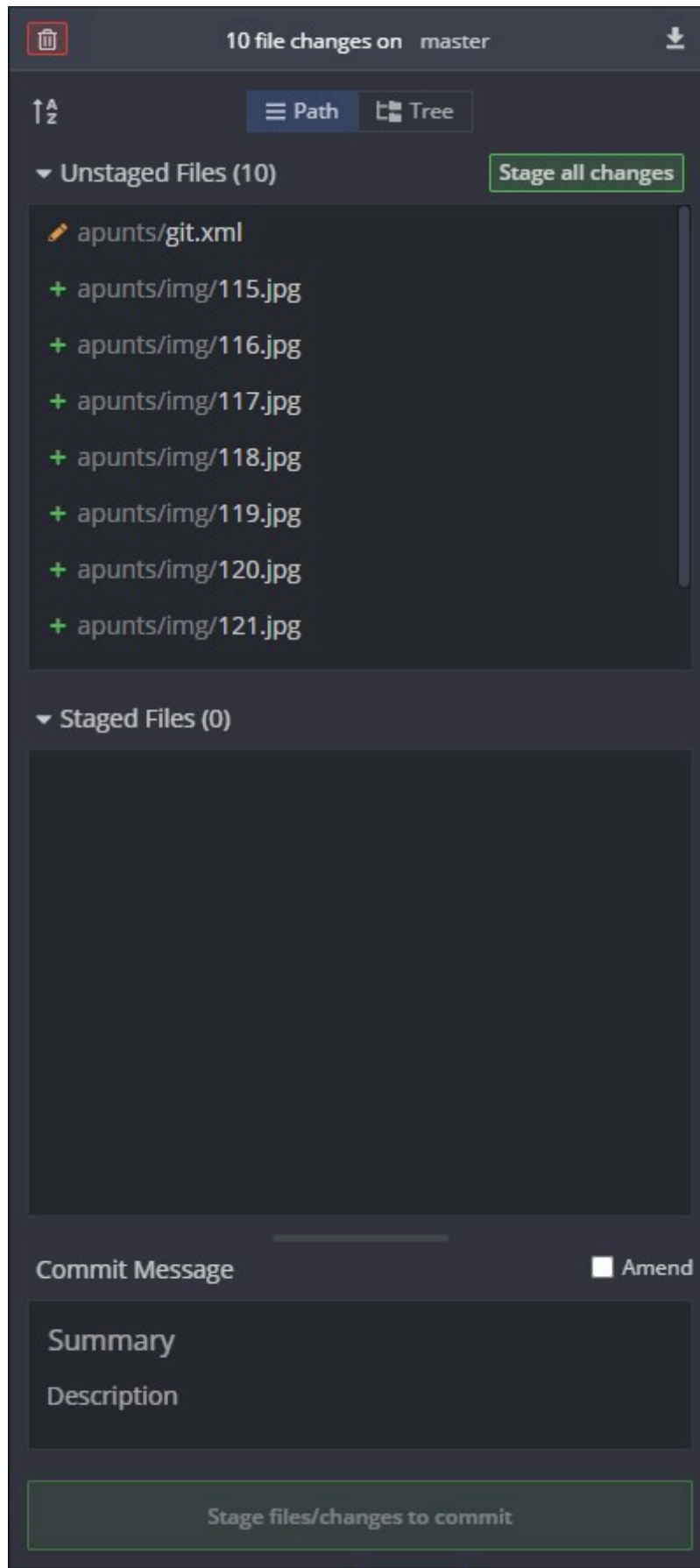
- Branques que hi ha en local i en remot



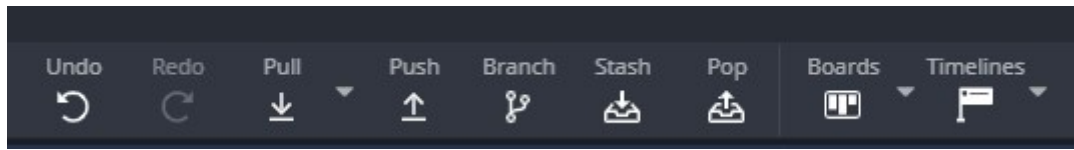
- Línia de temps del projecte: Esquema de commits, i branques que hem fet servir.



- Directori de treball.



- Barra d'eines.



Hi ha un video explicatiu en aquest enllaç <https://www.gitkraken.com/download/windows64>