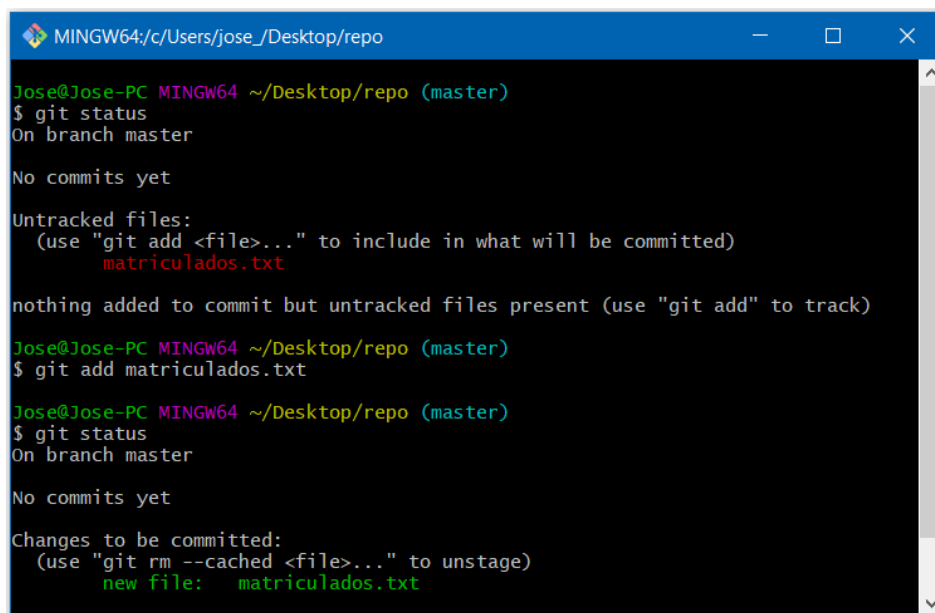


# Control de versiones

**Ejercicio 1.** Crea un fichero matriculados.txt, en cuya primera línea esté tu nombre y apellido. Realiza los siguientes pasos, indica en cada apartado el comando o comandos que has utilizado, así como razona y explica la respuesta a las preguntas que se piden en algunos apartados:

☐ Inicializa un repositorio git. ¿En qué estado está el fichero matriculados.txt?

- Primer creamos una carpeta, y escribimos consola dentro de esa carpeta con el comando "cd /..." e iniciamos la creación del repo con "git init". Con "git status" podemos verificar si en la carpeta master se ha incluido en la bbdd correctamente. Iniciamos git status y nos aparece como no seguido el archivo matriculados.txt, ahora tenemos que hacerle seguimiento con el comando "git add matriculados.txt". Verificamos de nuevo y ya aparece en verde como incluido en el repositorio.



```
MINGW64/c/Users/jose_/Desktop/repo
Jose@Jose-PC MINGW64 ~/Desktop/repo (master)
$ git status
On branch master

No commits yet

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

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

Jose@Jose-PC MINGW64 ~/Desktop/repo (master)
$ git add matriculados.txt

Jose@Jose-PC MINGW64 ~/Desktop/repo (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
       new file:   matriculados.txt
```

☐ Haz un primer commit. ¿Ha cambiado el estado del fichero?

- "git commit -m "Primera version"". Con esto se envía el archivo sin modificar y no aparece nada en el estado, no hay cambios.

- ☐ Añade dos matriculados más. ¿Y ahora, en qué estado está el fichero?
  - "git status" aparece como modificado, los cambios no han sido añadidos a commit.
- ☐ Crea otra revisión sin pasar por el área de preparación.
  - Si lanzo "git commit -m "Segunda version"" sin hacer antes el comando add para añadir los cambios, no se subirá el archivo modificado, por lo que hay que hacer lo del paso anterior saltándonos la preparación con el comando "git add -a"
- ☐ Crea otro fichero llamado nomatriculados.txt e incluye el nombre de una persona.
  - "git status" y "git add nomatriculados.txt".
- ☐ Añade un nombre más al fichero matriculados.txt
  - "git status"
- ☐ ¿En qué estado se encuentran ambos ficheros?
  - Aparece como modificado, en color rojo, no seguido el archivo nomatriculados.txt, en cambio matriculados.txt está como nuevo archivo en verde, seguido.
- ☐ Crea una revisión únicamente con el fichero nomatriculados.txt
  - "git commit -m nomatriculados.txt"
- ☐ Añade al área de preparación el fichero matriculados.txt
  - "git show" para ver los cambios y "git diff" para ver lo que no está en preparación. Por lo tanto, he de marcar "git status" para ver el estado que aparece como modificado y enviarlo para el seguimiento con "git add matriculados.txt"
- ☐ Añade un nuevo nombre a la lista de matriculados.txt
  - "git status"

☐ Saca el fichero matriculados.txt del área de preparación sin perder al último matriculado.txt

- `"git rm --cached matriculados.txt"` para forzar el borrado, le añadimos `-f`.

☐ Haz una revisión que incluya a los dos últimos matriculados.

- `>"git diff", "git status" y "git commit -m "..."`

☐ Consulta el histórico de cambios

- `"git log"`

☐ Consulta qué se hizo en el segundo commit del fichero matriculados.txt

- Tras el comando anterior he de introducir `"git show 6b31444 (código revisión commit)"`.

☐ Borra el fichero matriculados.txt de la copia de trabajo y recupéralo del repositorio

- `"git rm matriculados.txt"` y para recuperarlo `"git reset HEAD matriculados.txt"`

**Ejercicio 2.** Dentro de una carpeta llamada crea un fichero index.html, que tendrá un título h1 con el contenido textual "NOMBRE DE LA EMPRESA". Debajo del título incluye 3 párrafos con texto de relleno. Realiza los siguientes pasos indicando qué comando has ejecutado:

☐ Inicializa un repositorio git.

- `"cd /..."` e iniciamos la creación del repo con `"git init"`.

☐ Comprueba el estado del repositorio y coméntalo

- Con `"git status"` podemos verificar si en la carpeta master se ha incluido en la bbdd correctamente. Iniciamos `git status` y nos aparece como no seguido el archivo index.html

- ☐ Crea una primera revisión.
  - `"git add index.html"` y `"git commit -m """`.
- ☐ Comprueba el estado del repositorio y coméntalo.
  - `"git status"`, en el estado aparece como `"On branch master nothing to commit, working tree clean"`. No aparece nada en el estado, no hay cambios en el repositorio local.
- ☐ Añade una imagen de relleno (<https://picsum.photos>)
- ☐ Comprueba las diferencias entre la copia de trabajo y repositorio
  - `"git status"` donde aparece que no encuentra la imagen, se ha de insertar `git add` para añadirlo y `"git diff"` para indicarme que después del párrafo 3 ha habido cambios que no han sido seguidos.
- ☐ Añade el fichero al área de preparación
  - `"git add index.html"`, `"git add imagen.jpg"` para añadir los archivos al área de preparación y `"git commit -m index,html e imagen.jpg"` para confirmar el seguimiento en el repositorio local.
- ☐ Comprueba las diferencias entre área de preparación y repositorio
  - El staging área es un espacio de memoria ram donde están guardados los cambios, pero todavía no están en el repositorio, sin seguimiento. Repositorio es el lugar donde se realiza el seguimiento de los cambios, cuando algo está en el repositorio, ya está en la historia de git.
- ☐ Crea la revisión
  - `"git commit index.html"` y `"git status"`. `On branch master nothing to commit, working tree clean.`
- ☐ Crea una rama llamada empresaCoches
  - `"git checkout -b empresaCoches"`
- ☐ Incluye en esa rama un primer commit cambiando el contenido del título.
  - `"git add index.html"` se ha añadido a la nueva rama y `"git commit -m index.html"`

☐ En master crea otro commit con añadiendo dos párrafos más de texto de relleno

- `"git checkout master"` para ir a master, `"git add index.html"` y `"git commit -m index.html"` para guardar los dos párrafos.

☐ Desde empresaCoches crea un nuevo commit con dos párrafos más de texto de relleno pero diferentes de los puestos en la rama master

- `"git checkout empresaCoches"` para ir a la nueva rama, `"git add index.html"` y `"git commit -m index.html"` para guardar los dos párrafos en empresaCoches.

☐ Mezcla en empresaCoches la rama master de tal manera que aparezcan los párrafos de relleno de master ¿Qué ha pasado? Comenta esta situación.

>`"git checkout master"` para ir a master y `"git merge empresaCoches"` para fusionar ambas ramas. Aparece el siguiente mensaje:

```
Auto-merging index.html
```

```
CONFLICT (content): Merge conflict in index.html
```

```
Automatic merge failed; fix conflicts and then commit the result.
```

Se ha producido un conflicto y es necesario resolverlo manualmente editando los archivos y después marcarlos como fusionados con `"git add index.html"`. Luego podemos hacer `"git diff master empresaCoches"` para ver la diferencia de cambios entre ramas.