Cuaderno de ejercicios de git/github avanzado

[**Ejercicio 1: 1**](#_gzs66kozslm7)

[**Ejercicio 2 3**](#_6z04icns9t93)

[**Ejercicio 3 5**](#_w9c9kntlrf0t)

[**Ejercicio 4: ramas y fast forward 7**](#_2z9hd13joub9)

[**Ejercicio 5: ramas y auto-merge 9**](#_gsl1g4wlsbds)

[**Ejercicio 5B 13**](#_p4c5a2vbh7j)

[Ejercicio a 13](#_totrd0dd7mbn)

[Ejercicio b 13](#_p0gidpl3ijtq)

[Ejercicio c 13](#_8xsnpx5ewmi9)

[Ejercicio d 14](#_noc1nnsz8oly)

[**Ejercicio 6: conflicto! 14**](#_301cv6w6ct8c)

[**Ejercicio 7: eliminando commits 17**](#_wvva14nu6mon)

[**Ejercicio 8: stash en la misma rama 18**](#_gxdatbx1bmwf)

[**Ejercicio 9: stash en rama diferente 20**](#_41ynl8m4coy)

[**Ejercicio 10: remotos 23**](#_65l0ve7pryw)

[**Ejercicio 11: stash 24**](#_mjpnglsfiim5)

[**Ejercicio 12: rebase 24**](#_by2ldkg5qeau)

[**Ejercicio 13: 27**](#_5sm6ky6r3yzx)

# Ejercicio 1:

Crea una carpeta llamada exercise1, entra dentro de ella

**> md exercise1**

**> cd exercise1**

Conviertela en un proyecto de git

**> git init**

Comprueba qué versión de git tienes instalada

**> git --version**

¿Se ha creado alguna carpeta especial? Si, visualízala

**> dir /ah windows**

**> ls -a linux y mac**

Crea un fichero vacío llamado index.html

**> touch index.html linux y mac**

**> echo "" > index.html win**

Mira el estado de los archivos en git

(están en working directory)

**> git status**

**> git status -s modo simple**

**> modo simple y la rama**

Pasa al fichero index.html al staging area

working directory → staging area

**> git add index.html**

Vuelve a mirar el estado de tus archivos en git

**> git status**

**> git status -s modo simple**

**> modo simple y la rama**

Te has equivocado y quieres pasar el fichero index.html que está ahora en el staging area al working directory.

working directory ← staging area

**> git rm --cached index.html**

Crea otro fichero vacío llamado datos.txt

**> touch datos.txt linux y mac**

**> win**

Añade los 2 ficheros al staging area

working directory → staging area

**> git add index.html datos.txt**

Crea un fichero llamado info.json

Este archivo NO lo moverás al staging area

**>touch info.json linux y mac**

**> win**

Vuelve a mirar el estado de tus archivos en git

**> git status -s**

Mueve los archivos del staging area al estado de repository

working directory → staging area → repository area

**> git commit -m "muevo"**

Vuelve a mirar por último el estado de tus archivos en git

**> git status -s**

# 

# 

# Ejercicio 2

Crea una carpeta llamada exercise2, entra dentro de ella

**>md exercise2**

**>cd exercise2**

Conviertela en un proyecto de git

**>git init**

Crea un fichero index.html

**> touch index.html**

El primer requisito que te piden implementar es:

Modifica el fichero para que tenga un fondo azul claro y un titulo con la frase "Azul"

<body style="background-color: aqua">

<h1>Azul</h1>

</body>

Añade el fichero al staging area

**> git add index.html**

Añade al fichero al repository area, poniendo como mensaje de commit "background blue"

**> git commit -m**  "background blue"

Mira la rama en la que estas

**> git branch**

Mira el listado de commits que has realizado (solo has hecho uno)

**> git log**

Ahora te piden un cambio en la web.

Modifica el fichero index.html, ponle color rosa suave de fondo, y cambia el texto del título por "Rosa"

**> code index.html**

<body style="background-color: pink">

<h1>Rosa</h1>

</body>

Añade el fichero al staging area

**>git add index.html**

Añade al fichero al repository area, poniendo como mensaje de commit "color rosa"

**>git commit -m “color rosa”**

Mira el listado de commits que has realizado (tendrán que aparecer 2)

**> git log**

Te has equivocado en el texto que le has puesto al commit.

Cambia el texto del último commit por el texto "background pink"

**>git commit --amend -m "backgroun pink "**

Mira el listado de commits que has realizado (tendrán que aparecer 2)

**> git log**

Mira el listado de commits que has realizado, usa el modo corto en una sola línea

**> git log --oneline --decorate --graph**

Visualiza la página web que has creado en tu navegador

Hay muchas formas

**>**

# Ejercicio 3

Crea una carpeta llamada exercise3, entra dentro de ella

**>md exercise3**

**>cd exercise3**

Conviertela en un proyecto de git

**>git init**

Crea un fichero index.html

**> touch index.html**

El primer requisito que te piden implementar es:

Modifica el fichero para que tenga:

* un fondo verde claro
* un título con la frase "Green"

<body style="background-color: Green">

<h1>Green</h1>

</body>

Añade el fichero al staging area

**>git add index.html**

Añade al fichero al repository area, poniendo como mensaje de commit "background green"

**> git commit -m** "background green"

Añade ahora al fichero index.html, un párrafo con la frase "Esta página tiene un fondo verde."

<body style="background-color: Green">

<h1>Green</h1>

<p>Esta página tiene fondo verde.</p>

</body>

Añade los cambios al repository area, pero usando 1 solo comando

working directory → staging area → repository area

**> git commit -am "add parafo"**

Mira el listado de commits que has realizado (tendrán que aparecer 2)

**> git log**

Necesitas hacer algunos cambios en la web:

* Añade un nuevo parrafo al final con la frase: "página creada por Mi"
* Cambia el color de fondo por Violet
* Guarda los cambios

<body style="background-color: Violet">

<h1>Green</h1>

<p>Esta página tiene fondo violeta.</p>

<p>Página creada por Mi</p>

</body>

Te has dado cuenta que los cambios, desde el último commit, que has hecho son una basura. Quieres dejar el fichero como estaba desde el ultimo commit. Es decir, quieres deshacer todos los cambios que tienes en el working directory.

**>git checkout -- index.html**

Tu team leader pide:

1. cambies el texto del título Green por la frase "COLOR VERDE"
2. que elimines el párrafo que decía "Esta página tiene fondo verde."

<body style="background-color: Green">

<h1>COLOR VERDE</h1>

</body>

Visualiza los cambios que has realizado usando, usando un comando de git

**> git diff**

Añade los cambios al staging area

**> git add index.html**

Visualiza los cambios que has realizado usando, usando un comando de git

**> git diff –staged**

# Ejercicio 4: ramas y fast forward

Crea una carpeta llamada exercise4, entra dentro de ella

**>md exercise4**

**>cd exercise4**

Conviertela en un proyecto de git

**>git init**

Crea un fichero index.htm con el contenido

<body>

<h1>Mi título</h1>

<p>Esta es mi web</p>

</body>

Añade los datos al repository area, con la frase de commit "simple web"

**>git add index.html**

**> git commit -m "simple web"**

Visualiza tu página web en el navegador

**>**

Tu equipo de trabajo decide hacer cambios, ha llegado a la conclusión que:

1. Todo debería estar centrado en el navegador.
2. El fondo debería ser un degradado

Como no tienes muy claro cómo van a quedar esos cambios…, vas a añadir los cambios en una nueva rama.

Primero crea la rama llamada "feature01"

**>git branch feature01**

**>git checkout feature01**

Otra forma más rápida

**>git checkout -b feature01**

Añadimos los cambios necesarios al fichero index.html:

<style>

body {

display: flex;

flex-direction: column;

justify-content: center;

align-items: center;

height: 100vh;

background: rgb(2, 0, 36);

background: linear-gradient(145deg, rgba(2, 0, 36, 1) 0%, rgba(9, 9, 121, 1) 35%, rgba(0, 172, 255, 1) 100%);

}

</style>

<body>

<h1>Mi título.</h1>

<p>Esta es mi web</p>

</body>

Parece que los cambios te gustan.

Añade los cambios a repository area, pon como mensaje de commit "in the middle"

**> git commit -am "in the middle"**

Mira el estado de los archivos en git

**>git status**

Mira el listado de commit que has hecho

**> git log**

Después de enseñarle los cambios al team leader y dar su visto bueno, vamos a fusionar los cambios que has realizados en la nueva rama (**feature01**) a la rama master.

Primero pásate a la rama master

**> git checkout master**

Fusiona los cambios de feature01 a la rama master

**>git merge feature01**

Date cuenta que no hay ningún tipo de conflicto entre los cambios que había en la rama master y los cambios que has hecho tu en la rama feature01. Git podrá mezclarlos sin ningún problema. A esto se le llama **Fast Forward Merge**.

Como ya no necesitas la rama feature01, borrala.

**> git branch -d feature01**

# Ejercicio 5: ramas y auto-merge

Crea una carpeta llamada exercise5, entra dentro de ella

**>md exercise5**

**>cd exercise5**

Conviertela en un proyecto de git

**>git init**

Crea un fichero index.htm con el contenido

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>web</title>

</head>

<body>

<h1>Esto es un título</h1>

</body>

</html>

Añade los datos al repository area, con la frase de commit "first commit"

**> git add index.html**

**> git commit -m “ first commit”**

Visualiza tu página web en el navegador

**>**

Crea una rama llamada feature02 y sitúate en ella

**> git checkout -b feature02**

Añade los siguientes estilos

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>web</title>

<style>

body{

background-color: darkmagenta;

color: #fff;

}

</style>

</head>

<body>

<h1>Esto es un título</h1>

</body>

borra

Añade los cambios al repository area, con el texto de commit "add body styles"

**>>git commit -am "add body styles"**

Ahora añade más estilos:

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>web</title>

<style>

body{

background-color: darkmagenta;

color: #fff;

}

h1{

font-size: xx-large;

}

</style>

</head>

<body>

<h1>Esto es un título</h1>

</body>

Añade los cambios al repository area, con el texto de commit "add h1 styles"

**>git commit -am "add h1 styles"**

Ahora muévete a la rama master. Verás que los cambios que has añadido en la rama feature01 no están…

**>git checkout master**

Modifica el archivo index.html y añade un párrafo al final con el texto: "Esto es un párrafo"

Añade los cambios al repository area, con el texto de commit "add paragraph"

**>git commit -am "add paragraf “**

Visualiza la lista de log de la rama master

**>git log**

Visualiza la lista de log de todas las ramas

**> git log --all**

Visualiza la lista de log de todas las ramas y de forma decorada en una sola linea

**> git log --graph --decorate --oneline --all**

Vamos a mezclar los cambios hechos en la rama feature02 con los cambios realizado en la rama master

Ya que estamos situados en la rama master…

**> git merge feature02**

Date cuenta que no hay ningún tipo de conflicto entre los cambios que había en la rama master y los cambios que has hecho tu en la rama feature02. Git podrá mezclarlos sin ningún problema. A esto se le llama **Auto-merge / auto-fusionado**

Visualiza la lista de log de todas las ramas y de forma decorada en una sola linea

**> git log --graph --decorate --online --all**

Como el comando de log decorado es muy largo de escribir, crea un **alias** llamado slog que ejecute ese comando tan largo

**>git config --global alias.slog "log --graph --decorate --oneline --all"**

**>git slog**

Este último commit es muy importante vamos a ponerle una etiqueta

* Ponle de nombre v1.0
* Ponle de mensaje "Versión inicial"

**>git tag -a v1.0 -m "Version inicial"**

Muestra los tags que has creado

**>git tag**

Muestra la información de un tag en particular

**>git show v1.0**

No te convence el último tag que has creado, borralo

**>git tag -d v1.0**

# Ejercicio 5B

## Ejercicio a

Dibuja (bolitas, nombre ramas y head)

Hace commit

Hace commit

Hace commit

Crea rama testing y situate en ella

Cambia a rama master

Cambia a rama testing

Hace commit

Cambia a rama master

Hace commit

## Ejercicio b

Dibuja (bolitas, nombre ramas y head)

Hace commit

Hace commit

Hace commit

Crea rama issue

Ponte en la rama issue

Hace commit

Muevete a rama master

Crea rama hotfix

Muevete a la rama hotfix

Hacer commit

## Ejercicio c

Dibuja (bolitas, nombre ramas y head)

Hace commit

Hace commit

Crea rama hotfix y ponte en ella

Hace commit

Hace commit

Mergueo la rama hotfix a la rama master

¿Qué ha ocurrido? Fast forward

## Ejercicio d

Dibuja (bolitas, nombre ramas y head)

Hace commit

Hace commit

Hace commit

Crea rama hotfix y ponte en ella

Hace commit

Hace commit

Se mueve a rama master

Hace commit

Mergueo la rama hotfix a la rama master

Que ha ocurrido? mergue sin conflicto

# Ejercicio 6: conflicto!

Crea una carpeta llamada exercise6, entra dentro de ella

**>**

**>**

Conviertela en un proyecto de git

**>**

Crea un fichero index.htm con el contenido

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>GATOS</title>

</head>

<body>

<h1>Esto es un título</h1>

</body>

</html>

Añade los datos al repository area, con la frase de commit "**first commit**"

**>**

**>**

Visualiza tu página web en el navegador

**>**

Crea una rama llamada feature03 y sitúate en ella

**>**

Cambia el título con la frase: "Título cambiado en la feature03"

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>GATOS</title>

</head>

<body>

<h1>Título cambiado en la feature03</h1>

</body>

Añade los cambios al repository area, con el texto de commit "**change title feature03**"

**>**

Ahora muévete a la rama master. Verás que los cambios que has añadido en la rama feature03 no están…

**>**

Cambia el título con la frase: "Título cambiado en master"

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>GATOS</title>

</head>

<body>

<h1>Título cambiado en Master</h1>

</body>

Añade los cambios al repository area, con el texto de commit "**change title master**"

**>**

Visualiza la lista de log de todas las ramas y de forma decorada en una sola linea

**>**

Vamos a mezclar los cambios hechos en la rama feature03 con los cambios realizado en la rama master

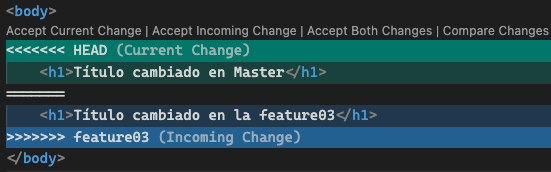
Ya que estamos situados en la rama master…

**>**

Date cuenta que ahora hay un problema. El título modificado en master y el título modificado en feature03 son distintos. ¿Cuál es que debería quedar? CONFLICTO!

Vamos a deshacer el conflicto.

Edita el fichero index.html y quédate con los cambios realizado en la feature03 (Incoming Change)



Añade los cambios al repository area, con el texto de commit "**resolv conflict**"

**>**

Visualiza la lista de log de todas las ramas y de forma decorada en una sola linea

**>**

# Ejercicio 7: eliminando commits

Crea una carpeta llamada exercise7, entra dentro de ella

>md exercise7

>cd exercise7

Conviertela en un proyecto de git

>git init

Crea un fichero index.html vacio

> touch index.html

Añade los datos al repository area, con la frase de commit "initial web"

> git add index.html

> git commit -m “initial web”

Ahora quieres añadir la foto de un perro. Descarga la foto de la url de https://picsum.photos/id/237/200/300

> curl -o dog.jpg https://picsum.photos/id/237/200/300

Añade los datos al repository area, con la frase de commit "initial web"

> git add dog.jpg

> git commit -m “initial web”

Te has dado cuenta que la foto tiene copyright, necesitas eliminar el último commit que has hecho

> git reset –soft HEAD ~1

# Ejercicio 8: stash en la misma rama

Crea una carpeta llamada exercise8, entra dentro de ella

**Crea una carpeta llamada exercise8, entra dentro de ella**

**>md exercise8**

**>cd exercise8**

**Conviertela en un proyecto de git**

**>git init**

Crea un fichero index.html con el siguiente contenido

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Tu web</title>

</head>

<body>

<h1>Esta es tu web</h1>

</body>

</html>

Añade los datos al repository area, con la frase de commit "**initial web**"

**> git add index.html**

**> git commit -m “initial web”**

Ahora añade los cambios que tu quieres a la página web

* más tags
* estilos
* lógica de negocio
* baja una foto de internet
* Añade la foto a la página web
* ….
* NO hagas aun el commit

De repente aparece el CEO de la empresa, te manda que dejes todo lo que estás haciendo, porque tienes que hacer un cambio urgente en la web.

Enfadado pero resignado, decides guardar todos los cambios en tu STASH para atender al CEO

**> git stash**

**Visualiza el estado de tu stash**

**> git stash list**

El CEO te pide que cambies el title del head con el texto: CEBEM

**>**

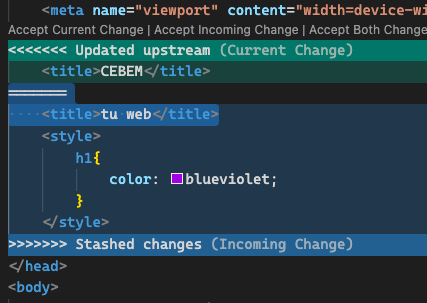
Añade los datos al repository area, con la frase de commit "**urgent title changes**"

**>**

El ceo se ha marchado contento, y tu ya puedes recuperar el trabajo que estabas haciendo.

**>**

Se ha producido un pequeño conflicto que hay que solucionar



Mezcla tus cambios con los cambios del CEO y comitea los cambios

**>**

# Ejercicio 9: stash en rama diferente

Crea una carpeta llamada exercise9, entra dentro de ella

**>**

**>**

Conviertela en un proyecto de git

**>**

Crea un fichero index.html con el siguiente contenido

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

</head>

<body style="background-color: palevioletred;">

<h1>PSA PEUGEOT CITROEN</h1>

</body>

</html>

Añade los datos al repository area, con la frase de commit "**initial web**"

**>**

**>**

Te ha tocado desarrollar una feature01 que añada un botón que al pulsarlo se muestre por pantalla la frase "Hola"

Para hacer estos cambios, te vas a cambiarte a la rama llamada "developer"

**>**

Mira las ramas que tienes creadas

**>**

Te pones a trabajar: añades el botón y la lógica de negocio necesaria…

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

</head>

<body style="background-color: palevioletred;">

<h1>PSA PEUGEOT CITROEN</h1>

<button onClick="alert('hola')">SALUDAR</button>

</body>

</html>

Añade los datos al repository area, con la frase de commit "**button added**"

**>**

**>**

Vas a seguir añadiendo más funcionalidades, en la feature02 tienes que añadir un formulario

<body style="background-color: palevioletred;">

<h1>PSA PEUGEOT CITROEN</h1>

<button onClick="alert('hola')">SALUDAR</button>

<form>

<input type="text" placeholder="nombre">

<input type="text" placeholder="apellido">

<input type="text" placeholder="email">

<input type="submit" value="enviar">

</form>

</body>

!!! De repente nuestro cliente llama enfadado diciendo que el título de la página web tendría que ser STELLANTIS y no el antiguo PSA PEUGEOT CITROEN. Piden de urgencia que dejes lo que estés haciendo y lo cambies.

No podemos merguear nuestro formulario ya que no lo tenemos un terminado.

Nos tenemos que mover a la rama master

ERROR **>**

No podemos movernos pues tenemos cambios pendientes. Tenemos 2 soluciones:

* Comitear los cambios (no es correcto pues el formulario no está terminado)
* Guardar los cambios en el stash

Vamos a guardar los cambios no terminados en el stash

**>**

Nos movemos a la rama master

**>**

Corregimos el título h1 por STELLANTIS

<body style="background-color: palevioletred;">

<h1>STELLANTIS</h1>

</body>

Añade los datos al repository area, Necesitas además añadir no solo un simple titulo al commit. Necesitas expresar el porqué de este cambio en varias lineas

**>**

Ya puedes continuar con tu trabajo. Vuelve a la rama developer

**>**

Recupera tu formulario guardado en el stash

**>**

Sigue trabajando en tu formulario y cuando estimes oportuno añade los datos al repository area, con la frase de commit "**form added**"

**>**

Merguea la rama develop a la rama master

**>**

**>**

Soluciona el conflicto y comitea los cambios

**>**

# Ejercicio 10: remotos

crea un repo en github proyecto1

#

clona ese repo en tu disco local

>

crea otro repo en github proyecto2

#

Visualiza los remotos que tiene tu repositorio

>

Añade un nuevo remoto a tu proyecto2

>

Borra el último remoto que has añadido

>

# Ejercicio 11: stash

próximamente…

# Ejercicio 12: rebase

Crea una carpeta llamada exercise12, entra dentro de ella

**>**

**>**

Conviertela en un proyecto de git

**>**

Crea un fichero index.htm con el contenido

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>web</title>

</head>

<body>

<h1>Título</h1>

</body>

</html>

Añade los datos al repository area, con la frase de commit "first commit"

**>**

**>**

Visualiza tu página web en el navegador

**>**

Crea una rama llamada feature12 y sitúate en ella

**>**

Añade los siguientes estilos

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>web</title>

<style>

body{

background-color: red;

color: #fff;

}

</style>

</head>

<body>

<h1>Título</h1>

</body>

Añade los cambios al repository area, con el texto de commit "add body styles"

**>**

Ahora añade más estilos:

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>web</title>

<style>

body{

background-color: red;

color: #fff;

}

h1{

font-size: 3em;

}

</style>

</head>

<body>

<h1>Título</h1>

</body>

Añade los cambios al repository area, con el texto de commit "add h1 styles"

**>**

Ahora muévete a la rama master. Verás que los cambios que has añadido en la rama feature12 no están…

**>**

Modifica el archivo index.html y añade un párrafo al final con el texto: "Esto es un párrafo"

Añade los cambios al repository area, con el texto de commit "add paragraph"

**>**

Visualiza la lista de log de la rama master

**>**

Visualiza la lista de log de todas las ramas

**>**

Visualiza la lista de log de todas las ramas y de forma decorada en una sola linea

**>**

Vamos a mezclar los cambios hechos en la rama feature12 con los cambios realizado en la rama master.

Pero vamos a usar la técnica rebase

**>**

**>**

Visualiza la lista de log de todas las ramas y de forma decorada en una sola linea

**>**

Haz que tanto la rama master como feature12 apunten al último commit lanzado

**>**

**>**

Vuelve a visualizar la lista de log de todas las ramas y de forma decorada en una sola linea

**>**

# Ejercicio 13:

Crea una carpeta llamada exercise13, entra dentro de ella

**>**

**>**

Conviertela en un proyecto de git

**>**

Crea un fichero index.htm con una pequeña página web

Añade los datos al repository area, con la frase de commit "**first commit**"

**>**

**>**

Visualiza tu página web en el navegador

**>**

Crea una rama llamada fix01 y sitúate en ella

**>**

Cambia el título con la frase: "Mi centro es Cebem"

Pon de color de fondo de la web el azul clarito con degradado.

Añade los cambios al repository area, con el texto de commit "**change title in fix01**"

**>**

Ahora muévete a la rama master.

**>**

Cambia el título con la frase: "Mi centro es CPR Cebem"

Añade los cambios al repositorio, con el texto de commit "**change title in master**"

**>**

Visualiza la lista de log de todas las ramas y de forma decorada en una sola línea

**>**

Vamos a mezclar los cambios hechos en la rama fix01 con los cambios realizado en la rama master

Ya que estamos situados en la rama master…

**>**

Soluciona el conflicto quedandote con los cambios realizados en master

>

Añade los cambios al repositorio,

>

Visualiza la lista de log de todas las ramas y de forma decorada en una sola línea

**>**