

Trabajar con ramas en Git: git branch

[Sistemas](#) [Git](#) Lectura: 17 min[🏠](#) > [Manuales](#) > [Manual de Git](#) 17 de noviembre de 2016

Cómo trabajar con ramas en Git, como crear ramas con branch, como pasar de unas ramas a otras con checkout y como fusionar ramas con merge.

En el día a día del trabajo con Git una de las cosas útiles que podemos hacer es trabajar con ramas. Las ramas son caminos que puede tomar el desarrollo de un software, algo que ocurre naturalmente para resolver problemas o crear nuevas funcionalidades. En la práctica permiten que nuestro proyecto pueda tener diversos estados y que los desarrolladores sean capaces de pasar de uno a otro de una manera ágil.

Ramas podrás usar en muchas situaciones. Por ejemplo imagina que estás trabajando en un proyecto y quieres implementar una nueva funcionalidad en la que sabes que quizás tengas que invertir varios días. Posiblemente sea algo experimental, que no sabes si llegarás a incorporar, o bien es algo que tienes claro que vas a querer completar, pero que, dado que te va a ocupar un tiempo indeterminado, es posible que en medio de tu trabajo tengas que tocar tu código, en el estado en el que lo tienes en producción.

Usamos cookies para garantizar que obtengas la mejor experiencia en nuestro sitio web. [Más información](#)

[Permitir cookies](#)[Configuración de cookies](#)



Bajo el supuesto anterior lo que haces es crear una rama. Trabajas dentro de esa rama por un tiempo, pero de repente se te cuelga el servidor y te das cuenta que hay cosas en el proyecto que no están funcionando correctamente. Los cambios de la rama no están completos, así que no los puedes subir. En ese instante, lo que las ramas Git te permiten es que, lanzando un sencillo comando, poner de nuevo el proyecto en el estado original que tenías, antes de empezar a hacer esos cambios que no has terminado. Perfecto! solucionas la incidencia, sobre el proyecto original y luego puedes volver a la rama experimental para seguir trabajando en esa idea nueva.

Llegará un momento en el que, quizás, aquellos cambios experimentales los quieras subir a producción. Entonces harás un proceso de fusión entre la rama experimental y la rama original, operación que se conoce como merge en Git.

Las aplicaciones de las ramas son, como puedes imaginar, bastante grandes. Espero haber podido explicar bien una de ellas y que te hayas podido hacer una idea antes de seguir la lectura de este artículo.

Nota: Ten en cuenta que este artículo es un tanto avanzado, pensado para personas que ya trabajan con Git. Puedes aprender cosas más básicas que damos por sabidas aquí en el [Manual de Git](#).

Git branch

Usamos cookies para garantizar que obtengas la mejor experiencia en nuestro sitio web. Más información

[Permitir cookies](#)

[Configuración de cookies](#)

subcomandos de Git útiles y necesarios para trabajar con ramas, como `checkout` para moverse entre ramas o `merge` para fusionar ramas.

Puedes comenzar tu primera práctica para trabajar con ramas. Haremos algo tan sencillo como lanzar el comando `"git branch"` a secas. Esto nos dará el listado de ramas que tengamos en un proyecto. Pero hay que advertir que las ramas de un repositorio local pueden ser distintas de las ramas de un repositorio remoto. Por ejemplo, cuando clonas un repositorio de GitHub generalmente estás clonando únicamente la rama master y no todas las ramas que se hayan creado a lo largo del tiempo. Otro ejemplo es cuando creas una rama en tu repositorio local. En este caso la rama la tendrás simplemente en tu proyecto local y no se subirá al repositorio remoto hasta que no lo especifiques. Oviamente, todas estas cosas, y otras, son las que vamos a ver en este artículo.

La rama master

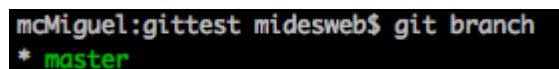
Cuando inicializamos un proyecto con Git automáticamente nos encontramos en una rama a la que se denomina "master".

Puedes ver las rama en la que te encuentras en cada instante con el comando:

```
git branch
```

Esta rama es la principal de tu proyecto y a partir de la que podrás crear nuevas ramas cuando lo necesites.

Si has hecho algún commit en tu repositorio observarás que después de lanzar el comando `"git branch"` nos informa el nombre de la rama como "master".



```
mcMiguel:gittest midesweb$ git branch
* master
```

Usamos cookies para garantizar que obtengas la mejor experiencia en nuestro sitio web. Más información

[Permitir cookies](#)

[Configuración de cookies](#)

Crear una rama nueva

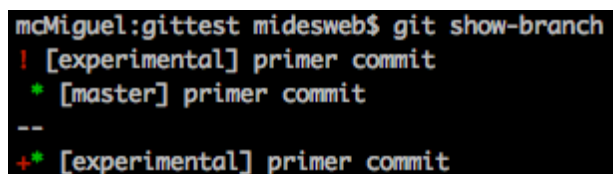
El procedimiento para crear una nueva rama es bien simple. Usando el comando `branch`, seguido del nombre de la rama que queremos crear.

```
git branch experimental
```

Este comando en sí no produce ninguna salida, pero podrías ver las "branches" de un proyecto con el comando `"git branch"`, u obtener una descripción más detallada de las ramas con este otro comando:

```
git show-branch
```

Esto nos muestra todas las ramas del proyecto con sus commits realizados. La salida sería como la de la siguiente imagen.



```
mcMiguel:gittest midesweb$ git show-branch
! [experimental] primer commit
* [master] primer commit
--
+* [experimental] primer commit
```

Pasar de una rama a otra

Para moverse entre ramas usamos el comando `"git checkout"` seguido del nombre de la rama que queremos que sea la activa.

```
git checkout experimental
```

esta sencilla operación tiene mucha potencia, porque nos cambiará automáticamente todos los archivos de nuestro proyecto, los de todas las carpetas, para que tengan el contenido en el que se encuentren en la correspondiente rama.

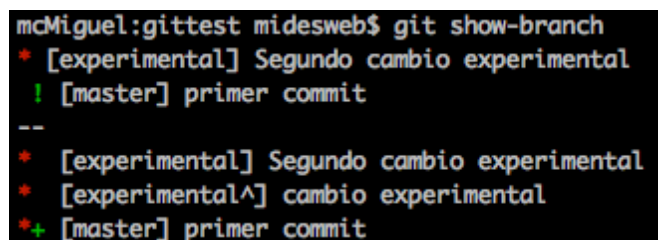
Usamos cookies para garantizar que obtengas la mejor experiencia en nuestro sitio web. Más información

[Permitir cookies](#)

[Configuración de cookies](#)

cambios en los archivos.

Si estando en la rama experimental haces un par de commit, observarás que al hacer el `show-branches` te mostrará nuevos datos:

A terminal window showing the output of the 'git show-branch' command. The prompt is 'mcMiguel:gittest midesweb\$'. The output lists three branches: '[experimental] Segundo cambio experimental' (marked with a red dot), '[master] primer commit' (marked with a green exclamation mark), and a separator line '--'. Below the separator, it lists the same three branches again, but with different markers: '[experimental] Segundo cambio experimental' (red dot), '[experimental^] cambio experimental' (red dot), and '[master] primer commit' (green plus sign).

El comando `checkout` tiene la posibilidad de permitirte crear una rama nueva y moverte a ella en un único paso. Para crear una nueva rama y situarte sobre ella tendrás que darle un nombre y usar el parámetro `-b`.

```
git checkout -b otrarama
```

Como salida obtendrás el mensaje `Switched to a new branch 'otrarama'`. Eso quiere decir que, además de crear la rama, nuestra cabecera está apuntando hacia esta nueva branch.

Si te dedicas a editar tus ficheros, crear nuevos archivos y demás en las distintas ramas entonces podrás observar que al moverte de una a otra con `checkout` el proyecto cambia automáticamente en tu editor, mostrando el estado actual en cada una de las ramas donde te estás situando. Es algo divertido y, si eres nuevo en Git verás que es una magia que resulta bastante sorprendente.

Como estás entendiendo, el proyecto puede tener varios estados en un momento dado y tú podrás moverte de uno a otro con total libertad y sin tener que cambiar de carpeta ni nada parecido. Si usas un programa de interfaz gráfica de Git, como SourceTree o cualquier otro, podrás ver las ramas en un esquema gráfico más entendible que en la consola de comandos.

Usamos cookies para garantizar que obtengas la mejor experiencia en nuestro sitio web. Más información

[Permitir cookies](#)

[Configuración de cookies](#)

Fusionar ramas

A medida que crees ramas y cambies el estado de las carpetas o archivos tu proyecto empezará a divergir de una rama a otra. Llegará el momento en el que te interese fusionar ramas para poder incorporar el trabajo realizado a la rama master.

El proceso de fusionado se conoce como "merge" y puede llegar a ser muy simple o más complejo si se encuentran cambios que Git no pueda procesar de manera automática. Git para procesar los merge usa un antecesor común y comprueba los cambios que se han introducido al proyecto desde entonces, combinando el código de ambas ramas.

Para hacer un merge nos situamos en una rama, en este caso la "master", y decimos con qué otra rama se debe fusionar el código.

El siguiente comando, lanzado desde la rama "master", permite fusionarla con la rama "experimental".

```
git merge experimental
```

Un merge necesita un mensaje, igual que ocurre con los commit, por lo que al realizar ese comando se abrirá "Vim" (o cualquier otro editor de consola que tengas configurado) para que introduzcas los comentarios que juzgues oportuno. Salir de Vim lo consigues pulsando la tecla ESC y luego escribiendo ":q" y pulsando enter para aceptar ese comando. Esta operativa de indicar el mensaje se puede resumir con el comando:

```
git merge experimental -m 'Esto es un merge con mensaje'
```

En la siguiente imagen puedes ver una secuencia de comandos y su salida. Primero el cambio a la rama master "git checkout master" luego el "git branch"

Usamos cookies para garantizar que obtengas la mejor experiencia en nuestro sitio web. Más información

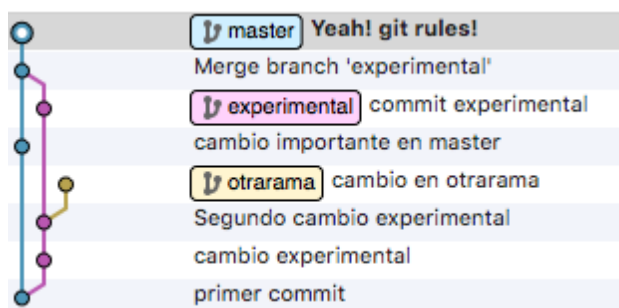
Permitir cookies

Configuración de cookies

```
mcMiguel:gittest midesweb$ git checkout master
Switched to branch 'master'
mcMiguel:gittest midesweb$ git branch
  experimental
* master
  otrarama
mcMiguel:gittest midesweb$ git merge experimental
Auto-merging miarchivo.txt
Merge made by the 'recursive' strategy.
 miarchivo.txt | 4 ++++
 otro_archivo.txt | 1 +
 2 files changed, 5 insertions(+)
 create mode 100644 otro_archivo.txt
```

Luego podremos comprobar que nuestra rama master tiene todo el código nuevo de la rama experimental y podremos hacer nuevos commits en master para seguir el desarrollo de nuestro proyecto ya con la rama principal, si es nuestro deseo.

Si tenemos un programa de Git por interfaz gráfica podremos ver el diagrama con el combinado de las ramas.



Subir una rama al repositorio remoto (Github o similares)

Como habíamos dicho anteriormente, por mucho que hagas la operativa descrita para crear ramas en tu ordenador, y las puedas ver en tu repositorio local con `git branch`, las ramas no se publicarán en Github o cualquier otro hosting de repositorios remoto. Para que esto ocurra tienes que realizar específicamente la acción de subir una rama determinada.

Usamos cookies para garantizar que obtengas la mejor experiencia en nuestro sitio web. Más información

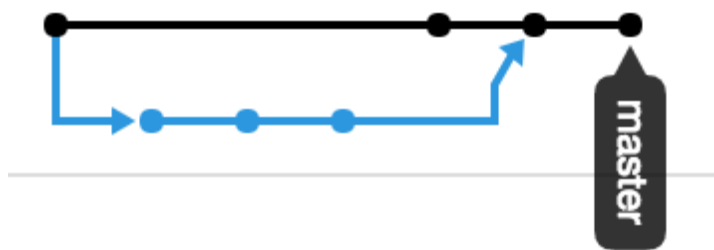
[Permitir cookies](#)

[Configuración de cookies](#)

```
git push -u origin experimental
```

Así estamos haciendo un push, empujando hacia origin (que es el nombre que se suele dar al repositorio remoto), la rama con nombre "experimental".

Por cierto, si subimos el proyecto a Github podremos ver también un diagrama de las ramas que hemos ido creando y fusionando a master, en la sección Graps / Network.



Borrar una rama

En ocasiones puede ser necesario eliminar una rama del repositorio, por ejemplo porque nos hayamos equivocado en el nombre al crearla. Aquí la operativa puede ser diferente, dependiendo de si hemos subido ya esa rama a remoto o si todavía solamente está en local.

Borrado de la rama en local

Esto lo conseguimos con el comando `git branch`, solamente que ahora usamos la opción `-d` para indicar que esa rama queremos borrarla.

```
git branch -d rama_a_borrar
```

Sin embargo, puede que esta acción no nos funcione porque hayamos hecho cambios que no se hayan salvado en el repositorio remoto, o no se hayan

Usamos cookies para garantizar que obtengas la mejor experiencia en nuestro sitio web. [Más información](#)

[Permitir cookies](#)

[Configuración de cookies](#)


```
git branch -D rama_a_borrar
```

Debes prestar especial atención a esta opción "-D", ya que al eliminar de este modo pueden haber cambios que ya no se puedan recuperar. Como puedes apreciar, es bastante fácil de confundir con "-d", opción más segura, ya que no permite borrado de ramas en situaciones donde se pueda perder código.

Eliminar un branch en remoto

Si la rama que queremos eliminar está en el repositorio remoto, la operativa es un poco diferente. Tenemos que hacer un push, indicando la opción --delete, seguida de la rama que se desea borrar.

```
git push origin --delete rama_a_borrar
```

Conclusión

Espero que estas notas te hayan ayudado a entender las ramas de Git y puedas experimentar por tu cuenta para hacer algunos ejemplos e ir cogiendo soltura. Realmente trabajar con ramas te dará la posibilidad de sacar mucho partido a Git y organizar mejor tus prepositorios, facilitando también el trabajo en grupo.

Para acabar te dejamos un enlace muy interesante y más detallado donde [aprender más del proceso de fusión de ramas](#).

En el [Manual de Git](#) encontrarás también muchas otras informaciones y tutoriales para extraer lo mejor de esta imprescindible herramienta.

Miguel Angel Alvarez

Miguel es fundador de DesarrolloWeb.com y la plataforma de formación online Escu...

Usamos cookies para garantizar que obtengas la mejor experiencia en nuestro sitio web. [Más información](#)

[Permitir cookies](#)[Configuración de cookies](#)

Manual de Git

Especificar versiones en
Git con tag

Pull Request con Git

¿Te ha resultado útil este artículo?

(0) 

 (0)

¿Alguna duda?

Pregunta y ayuda en la comunidad con tus respuestas
en la [sección de FAQ](#)

HACER UNA PREGUNTA

Aprender

Manuales

Temas

Comunidad

Preguntas y respuestas

Hacer una pregunta



Usamos cookies para garantizar que obtengas la mejor experiencia en nuestro sitio web. [Más información](#)

Permitir cookies

Configuración de cookies

[Home](#)

[Datos legales](#)

[Privacidad](#)

[Política de cookies](#)

[Contacto](#)

Usamos cookies para garantizar que obtengas la mejor experiencia en nuestro sitio web. [Más información](#)

Permitir cookies

Configuración de cookies