



git

Branches

Creando branches

Administrando branches

Branches

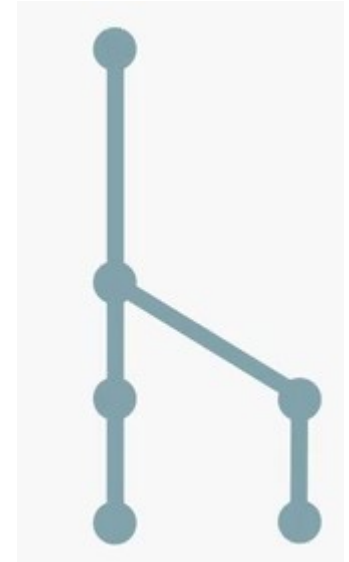
- Una de las características más potentes de los sistemas de control de versiones es el concepto de branching
- De hecho la facilidad con la que se realizan ramas en git es una de las causas de su popularidad
- La mayoría de los proyectos empiezan a ramificarse por las siguientes causas:
 - El cliente empieza a requerir cambios
 - Se detectan bugs
 - Puedes cambiar de opinión sobre las funcionalidades en las que trabajar
 - Puedes trabajar en equipo y cada uno estáis trabajando en diferentes partes del proyecto
 - etc.

Branches

- Imagina que estás en medio de un gran cambio en un proyecto y alguien reporta un bug crítico que necesita ser solucionado ahora mismo
- No puedes hacer un commit a los cambios que estabas realizando porque no están terminados
- Pero tampoco puedes esperar que estén terminados para solucionar el bug
- La solución es crear una nueva rama

Branches

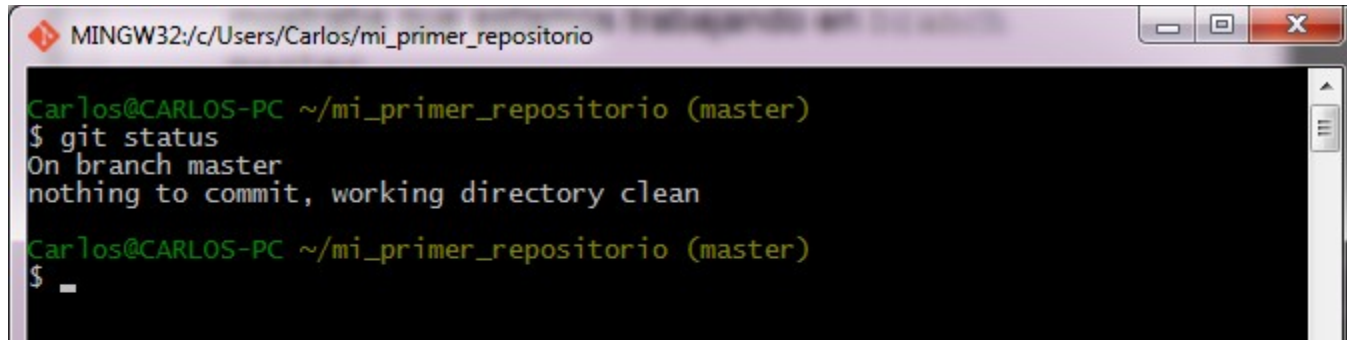
- Una rama o branch es un tipo de realidad alternativa de tu repositorio
- Las ramas te permiten tomar diferentes acciones en tu proyecto en paralelo
- Trabajar con ramas en un proyecto puede ser complicado y difícil en algunas ocasiones, pero git lo hace muy fácil



Creando branches

branch master

- Recuerda que cuando utilizamos el comando git status mostraba que estamos trabajando en
branch master

A screenshot of a terminal window titled 'MINGW32:/c/Users/Carlos/mi_primer_repositorio'. The terminal shows the following text:

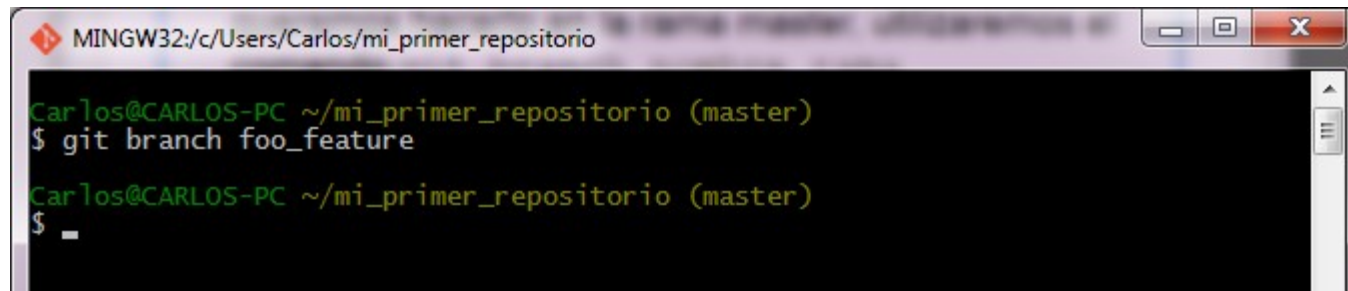
```
Carlos@CARLOS-PC ~/mi_primer_repositorio (master)
$ git status
On branch master
nothing to commit, working directory clean

Carlos@CARLOS-PC ~/mi_primer_repositorio (master)
$ _
```

- Master es la rama por defecto de un repositorio git, mucha gente lo ve como el tronco del repositorio
- Esta rama suele ser la versión canónica de tu proyecto que está en producción

Creando branches

- Si queremos crear una nueva característica y no queremos hacerlo en la rama master, utilizaremos el comando `git branch nombre_rama`
- El nombre de la rama no debería ser largo, intenta que sea lo más corto y claro posible



```
MINGW32:/c/Users/Carlos/mi_primer_repositorio

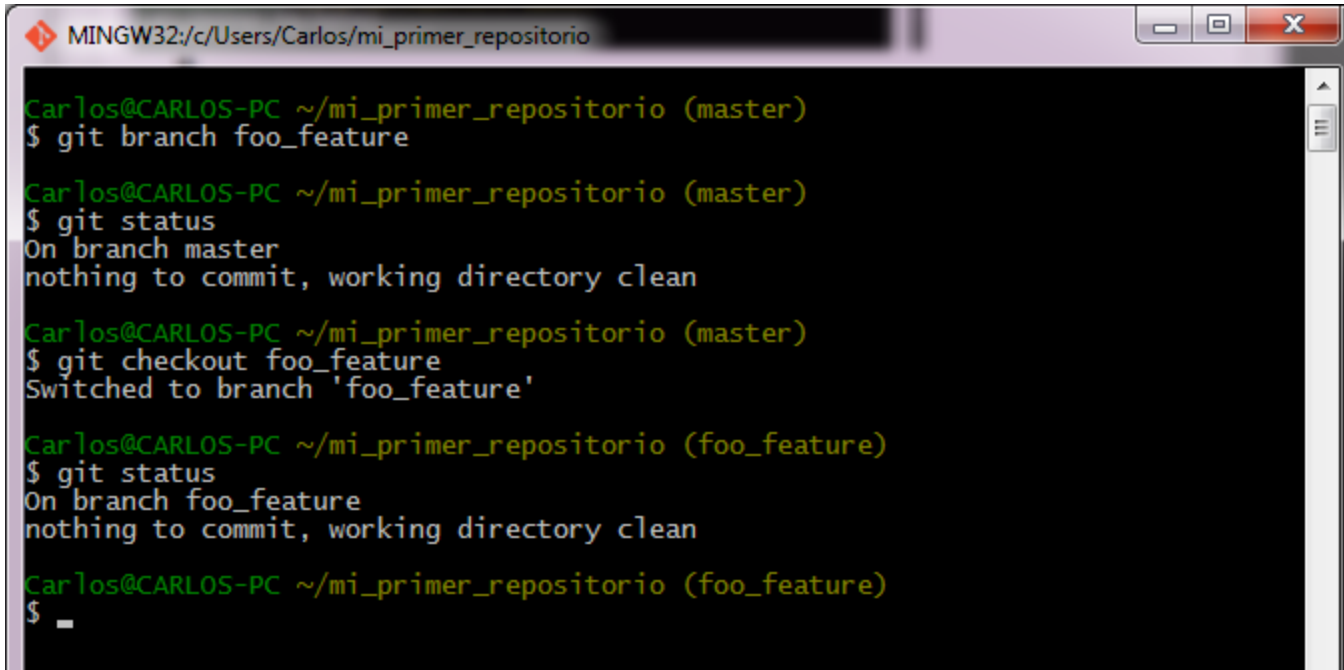
Carlos@CARLOS-PC ~/mi_primer_repositorio (master)
$ git branch foo_feature

Carlos@CARLOS-PC ~/mi_primer_repositorio (master)
$ _
```

Creando branches

Cambiar de rama

- Para cambiar de rama utilizaremos el comando `git checkout nombre_rama`

A screenshot of a terminal window titled 'MINGW32:/c/Users/Carlos/mi_primer_repositorio'. The terminal shows a series of Git commands and their outputs. The user starts on the 'master' branch, creates a new branch 'foo_feature', checks out 'foo_feature', and then checks the status again, confirming they are on the new branch with a clean working directory.

```
MINGW32:/c/Users/Carlos/mi_primer_repositorio

Carlos@CARLOS-PC ~/mi_primer_repositorio (master)
$ git branch foo_feature

Carlos@CARLOS-PC ~/mi_primer_repositorio (master)
$ git status
On branch master
nothing to commit, working directory clean

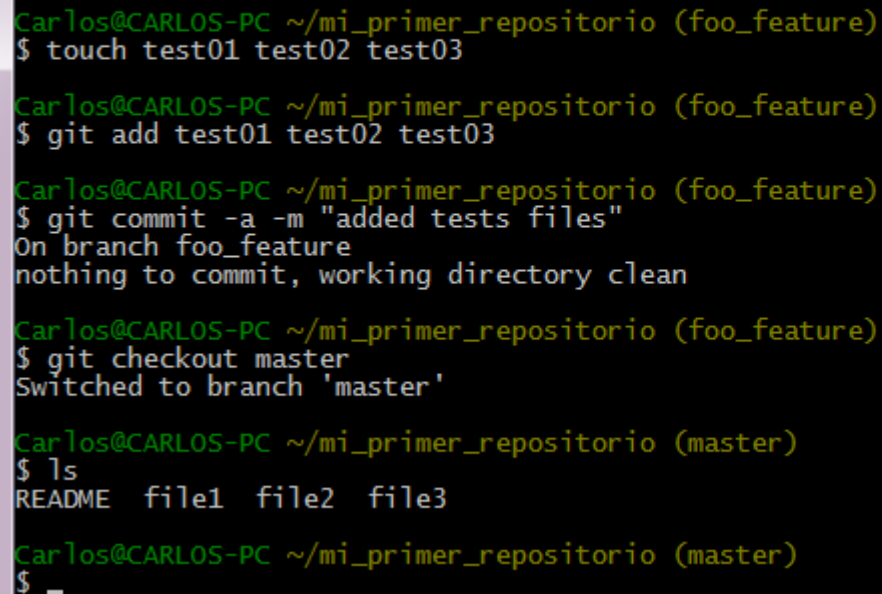
Carlos@CARLOS-PC ~/mi_primer_repositorio (master)
$ git checkout foo_feature
Switched to branch 'foo_feature'

Carlos@CARLOS-PC ~/mi_primer_repositorio (foo_feature)
$ git status
On branch foo_feature
nothing to commit, working directory clean

Carlos@CARLOS-PC ~/mi_primer_repositorio (foo_feature)
$ _
```


Creando branches

- Una vez que estamos en la rama podemos trabajar normalmente sin afectar a la rama master

A terminal window with a black background and green text. It shows a series of Git commands and their outputs. The user is in a branch named 'foo_feature' and creates three test files. They then attempt to commit, but Git reports that the working directory is clean. Finally, they checkout to the 'master' branch and list the files, which are README, file1, file2, and file3.

```
Carlos@CARLOS-PC ~/mi_primer_repositorio (foo_feature)
$ touch test01 test02 test03

Carlos@CARLOS-PC ~/mi_primer_repositorio (foo_feature)
$ git add test01 test02 test03

Carlos@CARLOS-PC ~/mi_primer_repositorio (foo_feature)
$ git commit -a -m "added tests files"
On branch foo_feature
nothing to commit, working directory clean

Carlos@CARLOS-PC ~/mi_primer_repositorio (foo_feature)
$ git checkout master
Switched to branch 'master'

Carlos@CARLOS-PC ~/mi_primer_repositorio (master)
$ ls
README  file1  file2  file3

Carlos@CARLOS-PC ~/mi_primer_repositorio (master)
$
```

- Para volver a la rama master, solo tenemos que utilizar el comando `git checkout master`

Creando branches

Crear una nueva rama y cambiarse

- Si queremos crear una nueva rama y cambiarnos en un solo comando utilizaremos la opción

-b

```
git checkout -b nombre_rama
```

Además, las nuevas ramas copian el contenido de la rama actual a la nueva

Creando branches

Crea una nueva rama en tu repositorio llamada new_feature

Cambia a la rama new_feature

Comprueba que estás en la rama new_feature

Vuelve a la rama master

Crea una nueva rama llamada another_new_feature y cambiate a esta con un solo comando

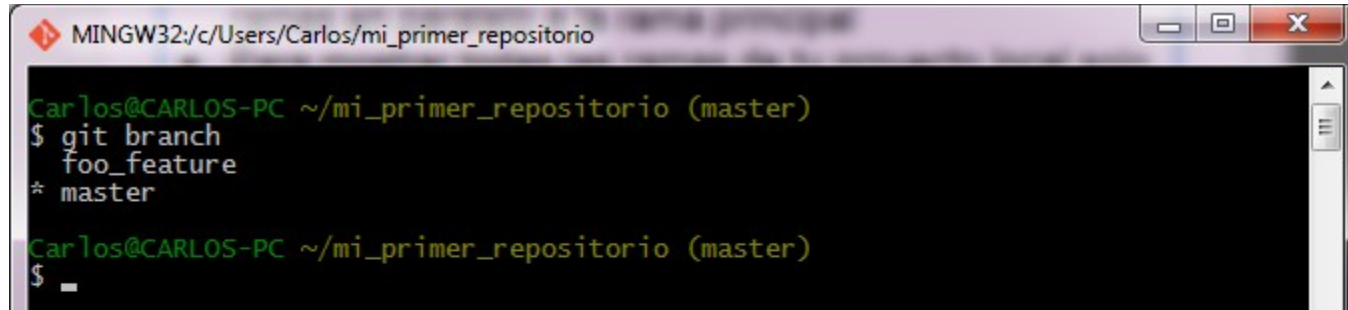
Administrando branches

Mostrar ramas

En proyecto grande puede haber decenas o cientos de ramas en paralelo a la rama principal

Para mostrar todas las ramas de tu proyecto local solo tienes que ejecutar el comando `git branch`

La rama marcada con un asterisco será tu rama actual

A screenshot of a terminal window titled 'MINGW32:/c/Users/Carlos/mi_primer_repositorio'. The terminal shows the command 'git branch' being executed, which lists two branches: 'foo_feature' and '* master'. The asterisk indicates that 'master' is the current branch. The prompt 'Carlos@CARLOS-PC ~/mi_primer_repositorio (master)' is visible at the top of the terminal output.

```
MINGW32:/c/Users/Carlos/mi_primer_repositorio
Carlos@CARLOS-PC ~/mi_primer_repositorio (master)
$ git branch
  foo_feature
* master

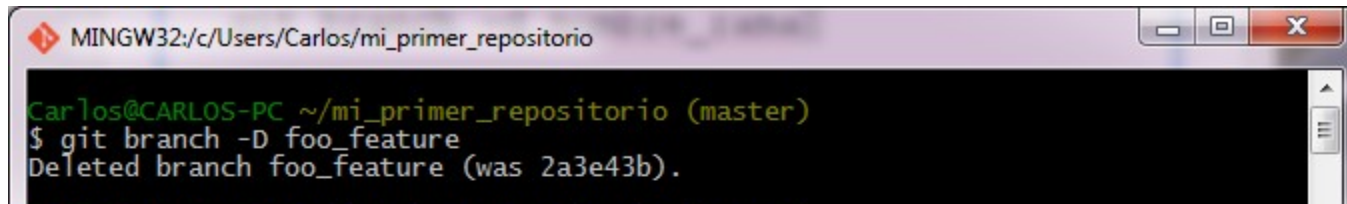
Carlos@CARLOS-PC ~/mi_primer_repositorio (master)
$
```

Administrando branches

Borrar ramas

Para borrar una rama se utiliza el la opción `-D`

```
git branch -D nombre_rama
```

A screenshot of a terminal window with a purple title bar. The title bar text is "MINGW32:/c/Users/Carlos/mi_primer_repositorio". The terminal content shows a prompt "Carlos@CARLOS-PC ~/mi_primer_repositorio (master)" followed by the command "\$ git branch -D foo_feature" and the output "Deleted branch foo_feature (was 2a3e43b).".

```
MINGW32:/c/Users/Carlos/mi_primer_repositorio  
Carlos@CARLOS-PC ~/mi_primer_repositorio (master)  
$ git branch -D foo_feature  
Deleted branch foo_feature (was 2a3e43b).
```

Administrando branches

Ejercicio

- ✓ Imprime todas las ramas de tu proyecto actual
- ✓ Borra la rama `another_new_feature`
- ✓ ¿para qué sirve la opción `git branch -d`? ¿Hace lo mismo que `git branch -D`?
- ✓ ¿Cómo puedes renombrar una rama?
- ✓ ¿Cómo podrías averiguar la rama de un commit con un identificador dado?

Webgrafia

<http://git-scm.com/>