Pro Git, el libro oficial de Git

2.5. Trabajando con repositorios remotos

Para poder colaborar en cualquier proyecto Git, necesitas saber cómo gestionar tus repositorios remotos. Los repositorios remotos son versiones de tu proyecto que se encuentran alojados en Internet o en algún punto de la red. Puedes tener varios, cada uno de los cuales puede ser de sólo lectura, o de lectura/escritura, según los permisos que tengas. Colaborar con otros implica gestionar estos repositorios remotos, y mandar (*push*) y recibir (*pull*) datos de ellos cuando necesites compartir cosas.

Gestionar repositorios remotos implica conocer cómo añadir repositorios nuevos, eliminar aquellos que ya no son válidos, gestionar ramas remotas e indicar si están bajo seguimiento o no, y más cosas. En esta sección veremos todos estos conceptos.

2.5.1. Mostrando tus repositorios remotos

Para ver qué repositorios remotos tienes configurados, puedes ejecutar el comando `git remote`. Mostrará una lista con los nombres de los remotos que hayas especificado. Si has clonado tu repositorio, deberías ver por lo menos "*origin*" — es el nombre predeterminado que le da Git al servidor del que clonaste —:

```
$ git clone git://github.com/schacon/ticgit.git
Initialized empty Git repository in /private/tmp/ticgit/.git/
remote: Counting objects: 595, done.
remote: Compressing objects: 100% (269/269), done.
remote: Total 595 (delta 255), reused 589 (delta 253)
Receiving objects: 100% (595/595), 73.31 KiB | 1 KiB/s, done.
Resolving deltas: 100% (255/255), done.
$ cd ticgit
$ git remote
origin
```

También puedes añadir la opción `-v`, que muestra la URL asociada a cada repositorio remoto:

```
$ git remote -v
origin  git://github.com/schacon/ticgit.git
```

Si tienes más de un remoto, este comando los lista todos. Por ejemplo, mi repositorio Grit tiene esta pinta:

```
$ cd grit
$ git remote -v
bakkdoor  git://github.com/bakkdoor/grit.git
cho45     git://github.com/cho45/grit.git
defunkt   git://github.com/defunkt/grit.git
koke      git://github.com/koke/grit.git
origin    git@github.com:mojombo/grit.git
```

Esto significa que podemos recibir contribuciones de cualquiera de estos usuarios de manera bastante fácil. Pero fíjate en que sólo el remoto origen tiene una URL SSH, por lo que es el único al que podemos enviar (veremos el por qué en el Capítulo 4).

2.5.2. Añadiendo repositorios remotos

Ya he mencionado y he dado ejemplos de repositorios remotos en secciones anteriores, pero a continuación veremos cómo añadirlos explícitamente. Para añadir un nuevo repositorio Git remoto, asignándole un nombre con el que referenciarlo fácilmente, ejecuta `git remote add [nombre] [url]`:

```
$ git remote
origin
$ git remote add pb git://github.com/paulboone/ticgit.git
$ git remote -v
origin  git://github.com/schacon/ticgit.git
pb  git://github.com/paulboone/ticgit.git
```

Ahora puedes usar la cadena "pb" en la línea de comandos, en lugar de toda la URL. Por ejemplo, si quieres recuperar toda la información de Paul que todavía no tienes en tu repositorio, puedes ejecutar `git fetch pb`:

```
$ git fetch pb
remote: Counting objects: 58, done.
remote: Compressing objects: 100% (41/41), done.
remote: Total 44 (delta 24), reused 1 (delta 0)
Unpacking objects: 100% (44/44), done.
From git://github.com/paulboone/ticgit
 * [new branch]      master     -> pb/master
 * [new branch]      ticgit     -> pb/ticgit
```

La rama maestra de Paul es accesible localmente como `pb/master` — puedes unirla a una de tus ramas, o copiarla localmente para inspeccionarla —.

2.5.3. Recibiendo de tus repositorios remotos

Como acabas de ver, para recuperar datos de tus repositorios remotos puedes ejecutar:

```
$ git fetch [remote-name]
```

Este comando recupera todos los datos del proyecto remoto que no tengas todavía. Después de hacer esto, deberías tener referencias a todas las ramas del repositorio remoto, que puedes unir o inspeccionar en cualquier momento. (Veremos qué son las ramas y cómo utilizarlas en más detalle en el Capítulo 3).

Si clonas un repositorio, el comando añade automáticamente ese repositorio remoto con el nombre de "origin". Por tanto, `git fetch origin` recupera toda la información enviada a ese servidor desde que lo clonaste (o desde la última vez que ejecutaste `fetch`). Es importante tener en cuenta que el comando `fetch` sólo recupera la información y la pone en tu repositorio local — no la une automáticamente con tu trabajo ni modifica aquello en lo que estás trabajando. Tendrás que unir ambos manualmente a posteriori —.

Si has configurado una rama para seguir otra rama remota (véase la siguiente sección y el Capítulo 3 para más información), puedes usar el comando `git pull` para recuperar y unir automáticamente la rama remota con tu rama actual. Éste puede resultarte un flujo de trabajo más sencillo y más cómodo; y por defecto, el comando `git clone` automáticamente configura tu rama local maestra para que siga la rama remota maestra del servidor del cual clonaste (asumiendo que el repositorio remoto tiene una rama maestra). Al ejecutar `git pul l`, por lo general se recupera la información del servidor del que clonaste, y automáticamente se intenta unir con el código con el que estás trabajando actualmente.

2.5.4. Enviando a tus repositorios remotos

Cuando tu proyecto se encuentra en un estado que quieres compartir, tienes que enviarlo a un repositorio remoto. El comando que te permite hacer esto es sencillo: `git push [nombre-remoto][nombre-rama]`. Si quieres enviar tu rama maestra (`master`) a tu servidor origen (`origin`), ejecutarías esto para enviar tu trabajo al servidor:

```
$ git push origin master
```

Este comando funciona únicamente si has clonado de un servidor en el que tienes permiso de escritura, y

nadie ha enviado información mientras tanto. Si tú y otra persona clonais a la vez, y él envía su información y luego envías tú la tuya, tu envío será rechazado. Tendrás que bajarte primero su trabajo e incorporarlo en el tuyo para que se te permita hacer un envío. Véase el Capítulo 3 para ver en detalle cómo enviar a servidores remotos.

2.5.5. Inspeccionando un repositorio remoto

Si quieres ver más información acerca de un repositorio remoto en particular, puedes usar el comando `git remote show [nombre]`. Si ejecutas este comando pasándole el nombre de un repositorio, como `origin`, obtienes algo así:

```
$ git remote show origin
* remote origin
  URL: git://github.com/schacon/ticgit.git
  Remote branch merged with 'git pull' while on branch master
    master
  Tracked remote branches
    master
    ticgit
```

Esto lista la URL del repositorio remoto, así como información sobre las ramas bajo seguimiento. Este comando te recuerda que si estás en la rama maestra y ejecutas `git pull`, automáticamente unirá los cambios a la rama maestra del remoto después de haber recuperado todas las referencias remotas. También lista todas las referencias remotas que ha recibido.

El anterior es un sencillo ejemplo que te encontrarás con frecuencia. Sin embargo, cuando uses Git de forma más avanzada, puede que `git remote show` muestre mucha más información:

```
$ git remote show origin
* remote origin
  URL: git@github.com:defunkt/github.git
  Remote branch merged with 'git pull' while on branch issues
    issues
  Remote branch merged with 'git pull' while on branch master
    master
  New remote branches (next fetch will store in remotes/origin)
    caching
  Stale tracking branches (use 'git remote prune')
    libwalker
    walker2
  Tracked remote branches
    acl
    apiv2
    dashboard2
    issues
    master
    postgres
  Local branch pushed with 'git push'
    master:master
```

Este comando muestra qué rama se envía automáticamente cuando ejecutas `git push` en determinadas ramas. También te muestra qué ramas remotas no tienes todavía, qué ramas remotas tienes y han sido eliminadas del servidor, y múltiples ramas que serán unidas automáticamente cuando ejecutes `git pull`.

2.5.6. Eliminando y renombrando repositorios remotos

Si quieres renombrar una referencia a un repositorio remoto, en versiones recientes de Git puedes ejecutar `git remote rename`. Por ejemplo, si quieres renombrar `pb` a `paul`, puedes hacerlo de la siguiente manera:

```
$ git remote rename pb paul
$ git remote
origin
paul
```

Conviene mencionar que esto también cambia el nombre de tus ramas remotas. Lo que antes era referenciado

en `pb/master` ahora está en `paul/master`.

Si por algún motivo quieres eliminar una referencia — has movido el servidor o ya no estás usando un determinado mirror, o quizás un contribuidor ha dejado de contribuir — puedes usar el comando `git remote rm`:

```
$ git remote rm paul
$ git remote
origin
```


Anterior

2.4. Deshaciendo cosas

Siguiente

2.6. Creando etiquetas

Indice de contenidos