Pro Git, el libro oficial de Git

2.1. Obteniendo un repositorio Git

Puedes obtener un proyecto Git de dos maneras. La primera toma un proyecto o directorio existente y lo importa en Git. La segunda clona un repositorio Git existente desde otro servidor.

2.1.1. Inicializando un repositorio en un directorio existente

Si estás empezando el seguimiento en Git de un proyecto existente, necesitas ir al directorio del proyecto y escribir:

```
$ git init
```

Esto crea un nuevo subdirectorio llamado .git que contiene todos los archivos necesarios del repositorio — un esqueleto de un repositorio Git. Todavía no hay nada en tu proyecto que esté bajo seguimiento. (Véase el Capítulo 9 para obtener más información sobre qué archivos están contenidos en el directorio `.git` que acabas de crear.)

Si deseas empezar a controlar versiones de archivos existentes (a diferencia de un directorio vacío), probablemente deberías comenzar el seguimiento de esos archivos y hacer una confirmación inicial. Puedes conseguirlo con unos pocos comandos `git add` para especificar qué archivos quieres controlar, seguidos de un `commit` para confirmar los cambios:


```
$ git add *.c
$ git add README
$ git commit -m 'versión inicial del proyecto'
```

Veremos lo que hacen estos comandos dentro de un minuto. En este momento, tienes un repositorio Git con

archivos bajo seguimiento, y una confirmación inicial.

2.1.2. Clonando un repositorio existente

Si deseas obtener una copia de un repositorio Git existente — por ejemplo, un proyecto en el que te gustaría contribuir — el comando que necesitas es `git clone`. Si estás familizarizado con otros sistemas de control de versiones como Subversion, verás que el comando es `clone` y no `checkout`. Es una distinción importante, ya que Git recibe una copia de casi todos los datos que tiene el servidor. Cada versión de cada archivo de la historia del proyecto es descargado cuando ejecutas `git clone`. De hecho, si el disco de tu servidor se corrompe, puedes usar cualquiera de los clones en cualquiera de los clientes para devolver al servidor al estado en el que estaba cuando fue clonado (puede que pierdas algunos *hooks* del lado del servidor y demás, pero toda la información versionada estaría ahí — véase el Capítulo 4 para más detalles —).

Puedes clonar un repositorio con `git clone [url]`. Por ejemplo, si quieres clonar la librería Ruby llamada Grit, harías algo así:

```
$ git clone git://github.com/schacon/grit.git
```

Esto crea un directorio llamado "grit", inicializa un directorio `.git` en su interior, descarga toda la información de ese repositorio, y saca una copia de trabajo de la última versión. Si te metes en el nuevo directorio `grit`, verás que están los archivos del proyecto, listos para ser utilizados. Si quieres clonar el repositorio a un directorio con otro nombre que no sea grit, puedes especificarlo con la siguiente opción de línea de comandos:

```
$ git clone git://github.com/schacon/grit.git mygrit
```

Ese comando hace lo mismo que el anterior, pero el directorio de destino se llamará mygrit.

Git te permite usar distintos protocolos de transferencia. El ejemplo anterior usa el protocolo `git://`, pero también te puedes encontrar con `http(s)://` o `usuario@servidor:/ruta.git`, que utiliza el protocolo de transferencia SSH. En el Capítulo 4 se introducirán todas las opciones disponibles a la hora de configurar el acceso a tu repositorio Git, y las ventajas e inconvenientes de cada una.

Anterior
Capítulo 2. Fundamentos de Git

Siguiente
2.2. Guardando cambios en el repositorio

Indice de contenidos

drupal

JavaScript

PHP

programación

Python

ruby

Symfony