Tutorial de Git

autor: Sócrates Díaz (Socrates_Dz)
twitter: @socrates_xD

En este pequeño tutorial explicaré qué es git, y cómo utilizarlo.

Git es un sistema de control de versiones (CVS). Es software libre y open source. Un sistema de control de versiones es un software para gestionar el historial de versiones de un proyecto. Una versión es el estado en el que se encuentra un proyecto en un momento dado.

Las ventajas de un CVS son copias de seguridad, deshacer cambios, historial de cambios, manejar diferentes versiones del proyecto, entre otros. Un proyecto puede gestionarse de manera individual o en grupo.

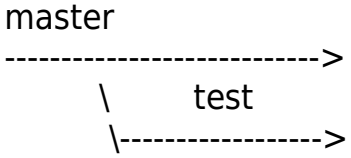
Git fue creado por Linus Torvalds en 2002. Después de ciertos problemas con un CVS creado por la compañía BitKeeper. BitKeeper no era lo suficientemente rápido para las necesidades del kernel como proyecto, muchos parches complicados tomaban hasta 30 segundos en aplicarse, sincronizar las versiones entre Linus Torvalds y Andrew Morton (mantenedor de la rama 2.6) demoraba 2 horas; se necesitaba algo veloz y distribuido, entonces Torvalds decidió crear un CVS llamado Git.

Algunos conceptos que debes manejar son:

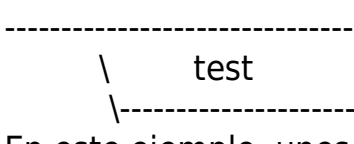
Repositorio: almacén de datos con el historial de versiones del proyecto.

Commit: cambios que introducimos en el proyecto.

Branch: crea una ramificación del proyecto. Ejemplo: tienes un branch que se llama master (**el cual de hecho es el branch por defecto que se usa en un proyecto**), pero quieres hacer pruebas partiendo del branch master sin afectar directamente al master, entonces haces un branch y lo llamas test.



Merge: consiste en unir branches. Ejemplo:



En este ejemplo, unes (o *merge*) el branch test con el master.

Push: consiste en subir los cambios que realizamos al proyecto. Es necesario crear un *commit* antes de hacer un *push*, sino no lo subes.

Clone: consiste en *clonar* un repositorio, es decir, copiar el repositorio a tu equipo.

Pull: es como *clone* pero solamente descarga los archivos que hayan sido actualizados o modificados, respecto a los que ya tienes.

Ahora pasemos a aprender cómo utilizarlo.

Para crear un proyecto:

Nota: el signo "$" que va antes del comando no tenemos que escribirlo en la terminal, ya que "$" es solo un indicador que nos dice que somos usuarios normales y no superusuarios. Y de hecho, si te fijas bien, el signo "$" ya está ahí por defecto.

1. Creas una carpeta con el nombre del proyecto (o entras en una ya existente):

   Creando la carpeta:

   ```
   $ mkdir proyecto
   ```

   Entrando en la carpeta:

   ```
   $ cd proyecto
   ```

2. Le dices a git que empiece a tomar nota de los cambios:

   ```
   $ git init
   ```

3. Agrega los archivos que quieras:

   ```
   $ git add <archivos>
   ```

Reemplaza `<archivos>` por los archivos que quieras agregar. Obviamente debe haber archivos en la carpeta o directorio. Para agregar todos los archivos del directorio pon un punto (.) . Ejemplo:

```
$ git add .
```

4. Haces un commit:

   ```
   $ git commit –m '<mensaje>'
   ```

   La opción -m se utiliza para escribir el mensaje directamente, en vez de abrir un editor de texto para escribirlo. En cuanto al mensaje, las comillas no se pueden obviar y el mensaje debe ser descriptivo, en este caso puedes poner 'Primera revisión', por ejemplo.

5. Le dices a github el repositorio al cual lo va a subir:

```
$ git remote add <nombre_repo> <repositorio>
```

Este paso básicamente consiste en ponerle un apodo o nombre al repositorio al que lo vamos a subir. Es una práctica común entre programadores poner como nombre *origin* al repositorio. El repositorio debe ser una dirección, por ejemplo: git://repos.com/usuario/mi_repo.git. Este paso sólo se hará una vez, así que no hay que hacerlo de nuevo, ya que git guardará ese nombre para referirse a tu repositorio.

Ejemplo:

```
$ git remote add origin git://repos.com/usuario/mi_repo.git
```

6. Subir los archivos al repositorio.

```
$ git push <nombre_repo> <branch>
```

Donde `<nombre_repo>` es el nombre del repositorio al que lo vamos a subir, y `<branch>` es el branch al que lo vamos a subir. El branch por defecto se llama master (aunque usted puede crear otro). Por ejemplo:

```
$ git push origin master
```

Donde `origin` es el nombre que le dimos al repositorio y `master` el branch al que lo subimos.

Nota.: Para que los comandos se ejecuten (después de que ejecute `git init)` usted debe estar en la carpeta del repositorio.