

GitHub Kata

Michell Pineda

Julian Arango

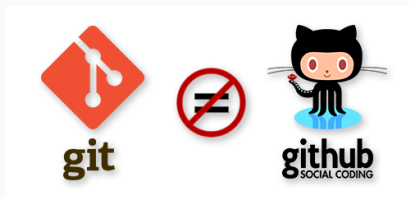
15 de Febrero de 2016

1. Diferencia entre Git y GitHub
2. Servidores de repositorios Git
3. Instalar Git
4. Cómo crear un repositorio?
5. Los tres estados
6. Cómo compartir un repositorio?
7. Committing, Pull y Push
8. Cómo crear un Branch?
9. Merging

Diferencia entre Git y Github

Git es un sistema de control de versiones distribuido, diseñado por Linus Torvalds, inicialmente para satisfacer las necesidades del Kernel de Linux, tras romper relaciones con la empresa que desarrollo su anterior herramienta (Bitkeeper).

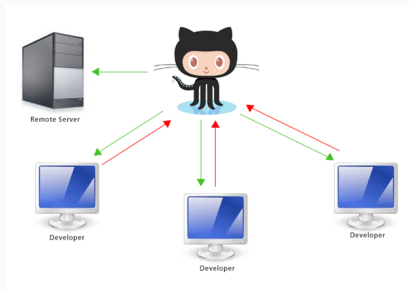
Y qué es Github? Es un hosting online para alojar repositorios que usan Git para el manejo del código fuente.



Servidores de repositorios Git

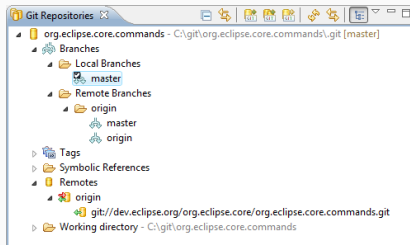
Algunos de ellos son:

- GitHub (Lo veremos más detallado)
- BitBucket
- GitLab
- Cloud Source Repositories (Beta)



Instalar Git

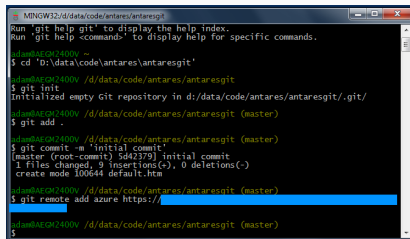
- Windows
 - Git Bash
 1. Ir a <http://git-scm.com/download/win>
 2. Instalar
 - Plug-in IDE
- Linux
 1. Entrar a la terminal
 2. Digitar “sudo apt-get install git-all”



Cómo crear un repositorio

1. Creamos el directorio (mkdir)
2. Desde la terminal nos ubicamos en el directorio (cd)
3. Digitamos "git init"

Si ya habamos creado el repositorio podemos clonarlo. Ubicados en el directorio, digitamos "git clone [URL del repositorio]".



```
MINGW32/d:/data/code/antares/antaresgit
Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

adam@AEQM2400V ~
$ cd 'D:/data/code/antares/antaresgit'

adam@AEQM2400V /d:/data/code/antares/antaresgit
$ git init
Initialized empty Git repository in d:/data/code/antares/antaresgit/.git/

adam@AEQM2400V /d:/data/code/antares/antaresgit (master)
$ git add .

adam@AEQM2400V /d:/data/code/antares/antaresgit (master)
$ git commit -m 'initial commit'
[master (root-commit) 5d42379] initial commit
1 files changed, 9 insertions(+), 0 deletions(-)
create mode 100644 default.htm

adam@AEQM2400V /d:/data/code/antares/antaresgit (master)
$ git remote add azure https://[redacted]

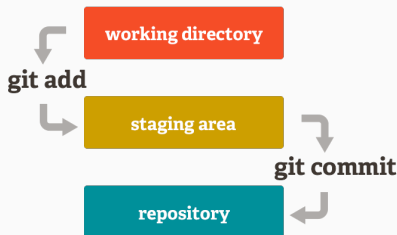
adam@AEQM2400V /d:/data/code/antares/antaresgit (master)
$
```

Los tres estados

Git tiene tres estados principales, que tus archivos pueden poseer:

- Committed: Guardado sin peligro en tu repositorio.
- Modified: El archivo tiene cambios, pero no se ha agregado.
- Staged: Agregado, pero no se ha hecho el commit al repositorio.

Esto nos conduce también a las tres principales secciones de Git:



Principales conceptos

- `git status`
- `git commit`
- `git add`
- `git log`
- `git branch`
- `reset`

Cómo compartir un repositorio?

Fork:

Es una copia exacta de un repositorio, sin embargo tendremos dos repositorios independientes, con diferente URL que pueden cada uno evolucionar de forma totalmente autónoma.

Repositorio upstream:

Es el repositorio original; no podremos hacer ningún cambio directamente sobre este repositorio, ya que no tendremos permisos de escritura sobre el mismo. Lo que haremos es crear nuestra propia copia (fork) del mismo.

Repositorio origin:

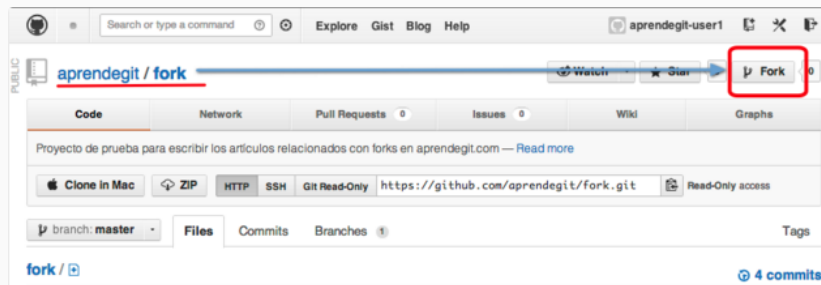
Es nuestra copia del repositorio que forkeamos alojada en github.
A este repositorio enviaremos los cambios que efectuaremos en nuestro repositorio local.

Repositorio local:

Es el repositorio local que tendremos en nuestra propia estación de trabajo.

1. Hacer un Fork

Del repositorio upstream, del que haremos copia, de esta forma ya tendremos nuestro repositorio origin.



2. Clonar repositorio origin en un repositorio local

```
1 $cd ~  
2 $mkdir <nombre carpeta>  
3 $cd <nombre carpeta>  
4 $git clone git@github.com:[tu_cuenta github]/[repositorio upstream].git
```

3. Guardar Cambios

```
5 $git status  
6 $git add  
7 $git commit -m "Subiendo modificaciones"  
8 $git push origin master
```

Mantener Repositorios Actualizados

Configurar el repositorio upstream como un repositorio remoto (remote) vinculado a nuestro repositorio local. Situados en el directorio del repositorio, deberemos ejecutar:

```
9 $git remote add upstream git@github.com:[tu_cuenta github]/[repositorio upstream].git
```

Cuando querramos traer las actualizaciones de upstream, deberemos ejecutar:

```
10 git fetch upstream  
11 git merge upstream/master
```

Committing, Pull y Push

Haciendo Commit

En GitHub, cuando se guardan cambios se le llama **commits**. Cada confirmación se le asociado un mensaje de confirmación, que es una descripción que explica por qu se hizo un cambio en particular.

```
1 $git commit -m "Mensaje"
```

Para obtener información sobre el estado de nuestro repositorio usaremos el comando **git status**.

```
2 |$git status
```


Para actualizar tu repositorio local al commit más nuevo, ejecuta **git pull** en tu directorio de trabajo para **bajar** y **fusionar** los cambios remotos. Al ejecutar **git pull**, 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.

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:

`git push [nombre-remoto][nombre-rama]`.

Si quieres enviar tu rama maestra (master) a tu servidor origen (origin), se ejecuta el comando `git push origin master` para enviar tu trabajo al servidor:

`git push origin master`

Cómo crear un Branch?

Cómo crear un branch?

y qué es un branch?. La traducción literal sería: rama. Es decir, dentro de nuestro sistema de control de versiones podemos ver el histórico de cambios como si de un árbol se tratase. De esta forma podemos ir abriendo ramas que parten bien de la rama principal (master) o de otra rama (branch).

```
1 $ git branch <nombre nuevo branch>
2 $ git checkout <nombre nuevo branch>|
```

Con esto creamos un branch y con el checkout nos movemos a él (el working directory queda apuntando a este branch para poder trabajar con el).

Visualiar los Branch que tenemos

Ejecutamos el comandos `git branch` y obtendremos una salida de este estilo:

```
1 $ git branch
2   master
3   *<nombre nuevo branch>|
```

Donde el `*` indica el branch activo. El listado anterior solo muestra los branch locales, pero también podemos listar los branch remotos si hacemos: `git branch -a`.

Branching y Merging

Para fusionar otra rama a tu rama activa (por ejemplo master), utiliza el comando `git merge [branch]`. En ambos casos git intentará fusionar automáticamente los cambios.

Desafortunadamente, no siempre será posible y se podrán producir conflictos. T eres responsable de fusionar esos conflictos manualmente al editar los archivos mostrados por git. Después de modificarlos, necesitas marcarlos como fusionados con el comando `git add [nombre archivo]`