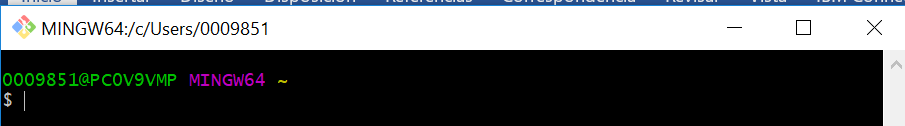
# 7 Manejo de GitHub

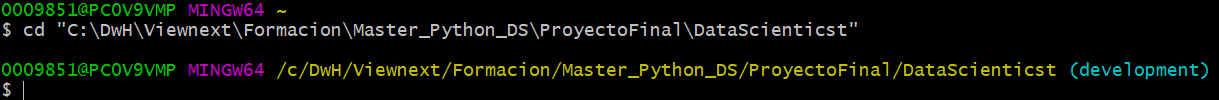
## Hacer subir código a git

Supongamos que hemos cambiado un código y queremos subir nuestros cambios a git. Para hacerlo desde consola sería necesario reaizar los siguientes pasos:

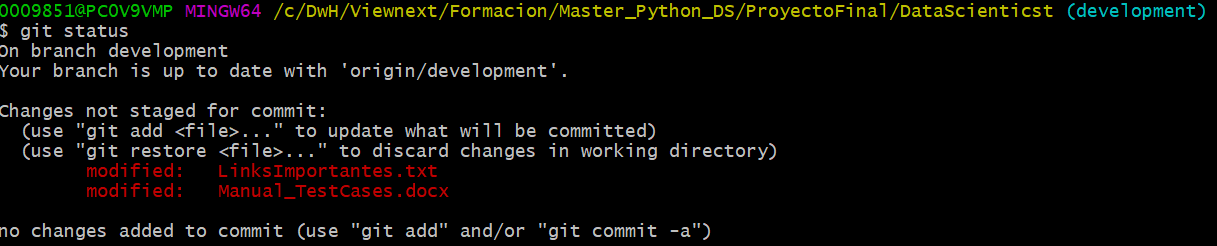
1. Abrir git bash para tener el entorno.



1. Ir a la ruta donde tenemos clonado el repositorio



1. Podemos hacer un git pull para traernos lo último que haya en el repositorio a nuestro local.
2. Hacemos un check status para ver los cambios existentes:

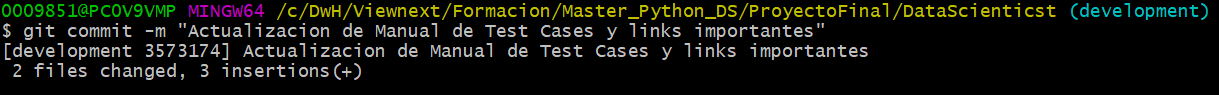


Esto nos indica que hay dos ficheros ( marcados en rojo ) que han sido modificados en nuestro local.

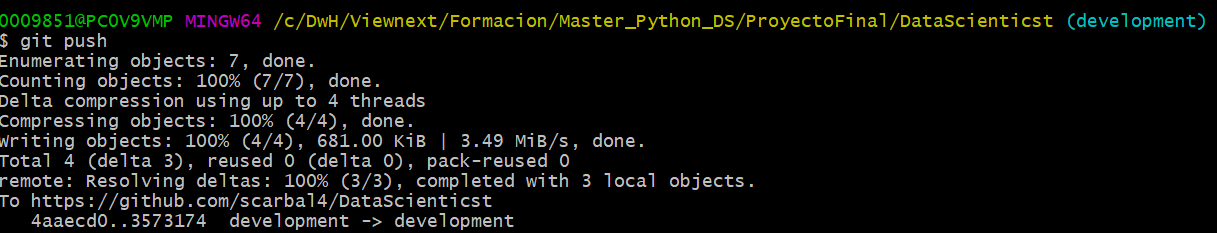
1. Tendremos que indicar que queremos añadir esos ficheros al repositorio. Git add . ( para añadir todos los ficheros y carpetas, si es uno en concreto sería git add “nombre\_fichero”):



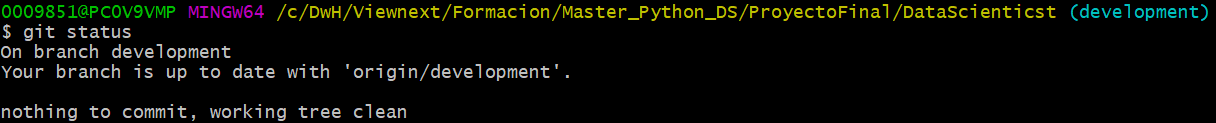
1. Ahora hay que comitear los cambios indicando una descripción de ayuda para futuro para saber que contiene esta versión:



1. Una vez hecho esto llega el momento de subirlo al repositorio, con git push



1. Para validar que no queda nada pendiente podríamos hacer un git status y ver que no hay cambios pendientes:



## Hacer un Git merge

Para poder mergear los cambios de una rama con otro los pasos a seguir son:

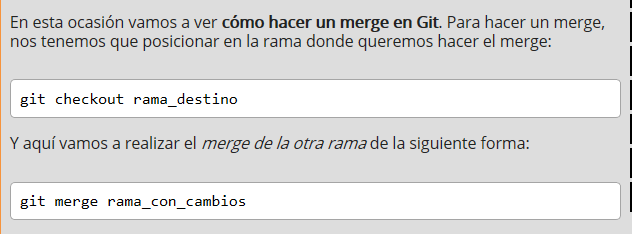
Git checkout rama destino ( normalmente master).

Ejemplo: git checkout master

Ahora ejecutamos el merge apuntando a la rama que contiene nuestros cambios ( por ejemplo develop)

Git merge develop

Si todo ha ido correctamente nos dirá que el merge ha sido resuelto.



## Cambiar entre cuentas de usuario en git

Via consola accedes al repositorio y usas los siguientes comandos:

git config user.name "Samuel"

git config user.email "scarbal4@corp.vodafone.es"

Con esto asignas estos valores **solamente** al repositorio en el que estás ubicado. Si lo que deseas es cambiar los datos de manera global, debes usar la opcion --global así:

git config --global user.name "Samuel"

git config --global user.email "scarbal4@gmail.com"

Ahora, para cambiar la configuracion del remoto puedes hacer esto: Primero lista el remoto que tienes configurado, cuya salida es algo similar a:

$ git remote -vv

origin https://username@github.com/owner/repo.git (fetch)

origin https://username@github.com/owner/repo.git (push)

Ahora puedes usar para eliminar el usuario:

$ git remote set-url origin https://github.com/owner/repo.git

O este para cambiar a otro usuario

$ git remote set-url origin <https://nuevousuario@github.com/owner/repo.git>

## Añadir un proxy a un repositiorio o a la cuenta de git

<https://gist.github.com/evantoli/f8c23a37eb3558ab8765>

git config --global http.proxy http://proxyUsername:proxyPassword@proxy.server.com:port

### Configure the proxy

You can configure these globally in your user ~/.gitconfig file using the --global switch, or local to a repository in its .git/config file.

#### Setting a global proxy

Configure a global proxy if all access to all repos require this proxy

git config --global http.proxy http://proxyUsername:proxyPassword@proxy.server.com:port

#### URL specific proxy

If you wish to specify that a proxy should be used for just some URLs that specify the URL as a git config subsection using http.<url>.key notation:

git config --global http.https://domain.com.proxy http://proxyUsername:proxyPassword@proxy.server.com:port

Which will result in the following in the ~/.gitconfig file:

[http]

[http "https://domain.com"]

proxy = http://proxyUsername:proxyPassword@proxy.server.com:port

Ejemplo:

Modificar el gitconfig del repositorio de convergence para añadirle el proxy:

C:\999-VFES-CloudAnalytics-Vodafone\convergence\.git

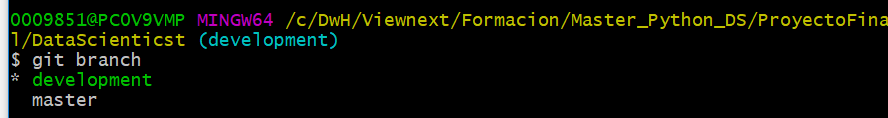
[https "https://github.vodafone.com"]

proxy = https://prxap15p.internal.vodafone.com:8080

## Trabajar con Ramas

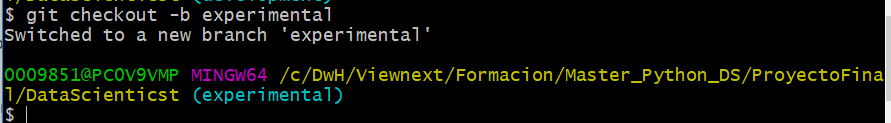
El trabajar con ramas es muy útil, puesto que nos permite tener los cambios de código únicamente afectados en esa rama y posteriormente fusionar esos cambios de código con la rama master para tener todo unificado.

Para ver las ramas disponibles en el repositorio ( en local ) se ejecutará el comando “git branch”



Para movernos de una rama a otra, como siempre se utilizará el comando “git checkout $Nombre\_Rama”

El comando “git checkout” tiene una opción “-b” que permite crear y moverse a una nueva rama:



Con el programa “SourceTree”, se podrían ver de manera gráfica las ramas del repositorio.

Para fusionar los cambios de una rama con otra existen dos formas de hacerlo ( siempre usando “git merge”

Forma 1:

Ubicarnos en la rama destino. Esto es, si queremos fusionar la rama experimental con la rama develop, en primer lugar tendremos que posicionarnos en la rama develop “git checkout develop” y posteriormente ahí hacer el merge

Git merge experimental

En este caso, se abrirá el editor VIM y pedirá que se introduzca un comentario.

Forma 2:

Ubicarnos en la rama destino ( a la que tiene los cambios y a la que queremos fusionar la información de la rama maestra):

Git checkout experimental

git merge master -m 'Un mensaje del merge de master en el branch experimental'

Con esto tenemos los cambios en nuestro repositorio local, pero es necesario tener los cambios en el repositorio remoto.

Para ello hay que empujar ( push ) con la opción –u:

git push -u origin experimental

Así estamos haciendo un push, empujando hacia origin (que es el nombre que se suele dar al repositorio remoto), la rama con nombre "experimental".

Si subimos el proyecto a Github, en la sección Graps / Network, podremos ver también un diagrama de las ramas que hemos ido creando y fusionando a master.

Si queremos borrar una rama en local, ejecutaremos:

git branch -d rama\_a\_borrar

Sin embargo, puede que esta acción no nos funcione porque hayamos hecho cambios que no se hayan salvado en el repositorio remoto, o no se hayan fusionado con otras ramas. En el caso que queramos forzar el borrado de la rama, para eliminarla independientemente de si se ha hecho el push o el merge, tendrás que usar la opción "-D".

git branch -D rama\_a\_borrar

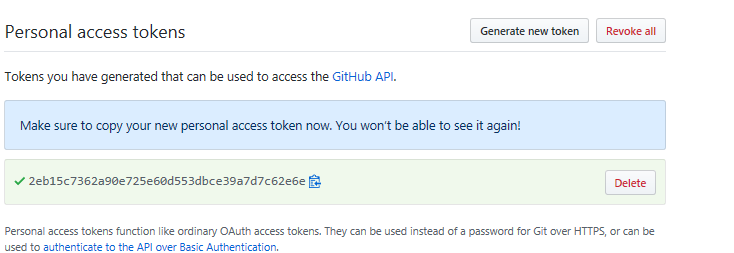
Si la rama que queremos eliminar está en el repositorio remoto, la operativa es un poco diferente. Tenemos que hacer un push, indicando la opción --delete, seguida de la rama que se desea borrar.

git push origin --delete rama\_a\_borrar

## Crear un Token para acceder a un repositorio privado

Supongamos que queremos acceder a un repositorio privado y que al intentar hacer un clone o un push nos muestra el siguiente mensaje:

Genremos un token personal dándole a todos los permisos y eso generará un código:



Ahora con ese código lo que suelen hacer en lugar de un clone es:

mkdir carpeta

**cd** carpeta

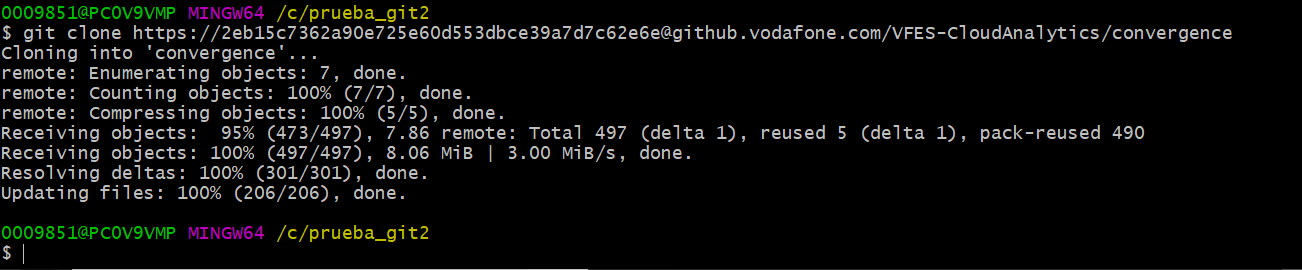
git init

git pull https://<token>@github.**com**/username/bar.git

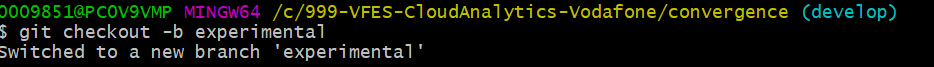
Ejemplo

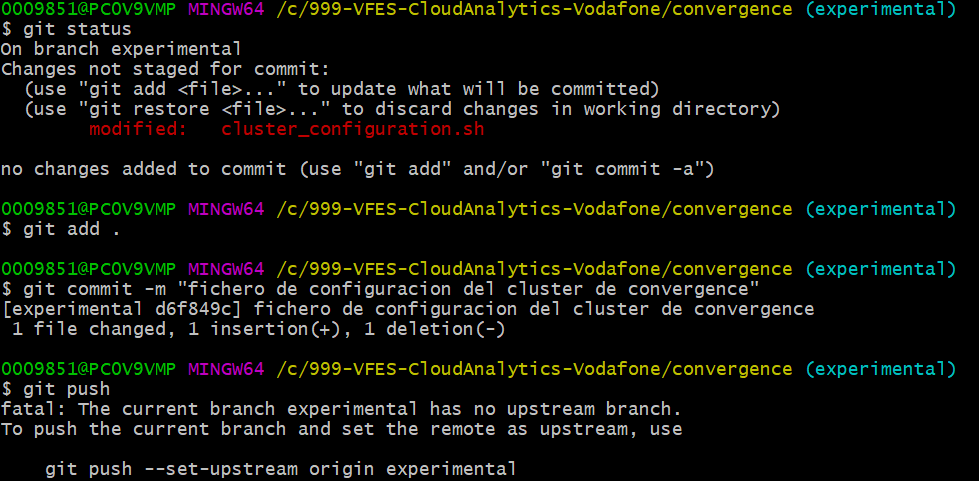
$ git pull <https://2eb15c7362a90e725e60d553dbce39a7d7c62e6e@github.vodafone.com/VFES-CloudAnalytics/convergence>

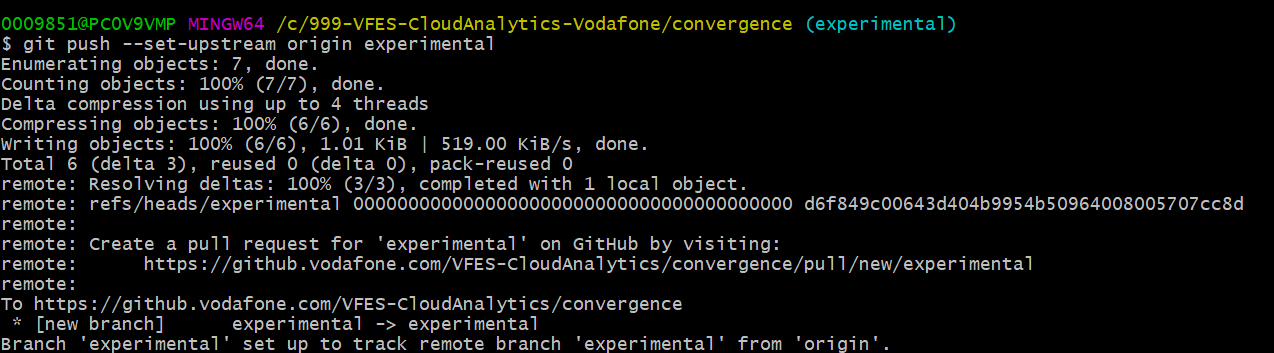
También vale para el git clone:



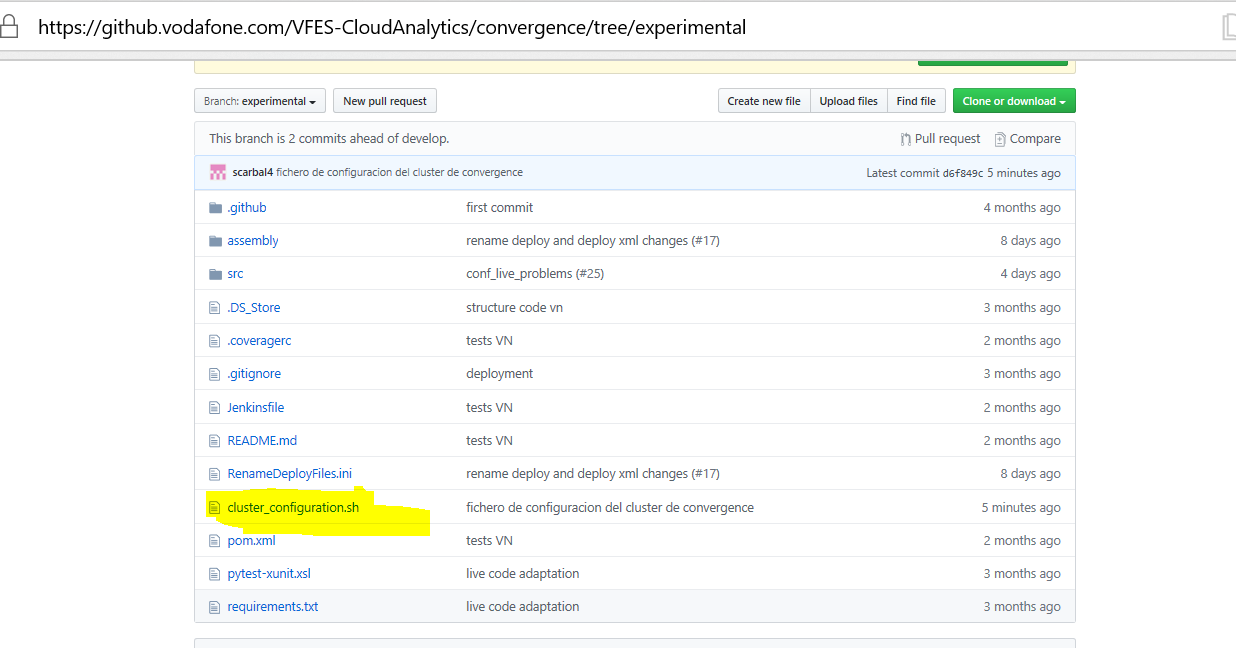
Ahora podríamos hacer un git push a la rama develop si tuviéramos permisos, como no lo tenía, lo que he hecho ha sido crear una rama y empujar los cambios de esa rama:



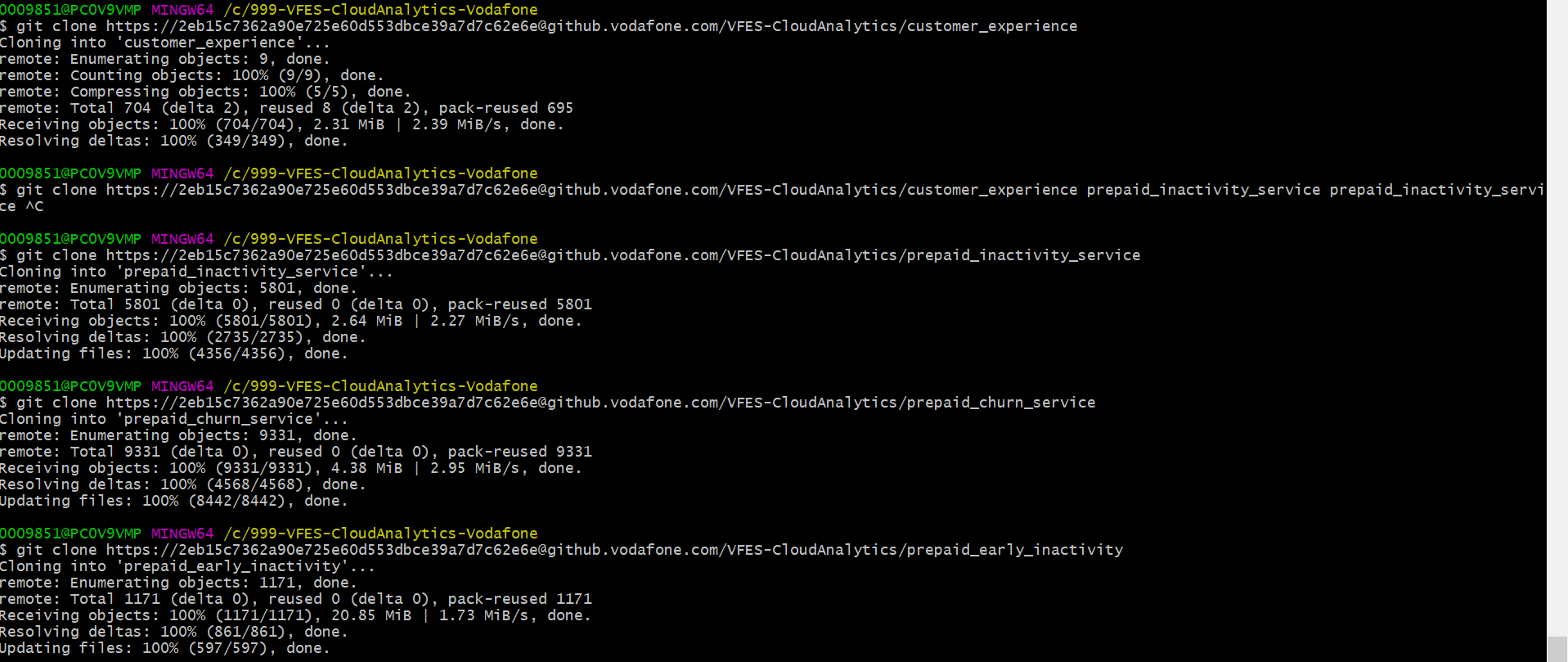




Y veriamos como está nuestra rama con el fichero subido



De esta forma, al ir clonando los repositorios:



Se van teniendo en la carpeta en local