**Zonas**

STASH | WORKSPACE | INDEX | LOCAL REPOSITORY | UPSTREAM REPOSITORY

**INICIAR EL REPOSITORIO**

git init

**WORKING DIRECTORY / WORKSPACE**

git add

**STAGING AREA / INDEX**

git commit

**LOCAL REPOSITORY**

git push / git pull

**UPSTREAM REPOSITORY**

**Configuración**

Esta configuración se puede encontrar en el archivo de configuración de git el **.gitconfig** que ese encuentra en la carpeta del usuario. También se puede hacer configuraciones para cada proyecto, esta configuración se guarda dentro de la carpeta .git del proyecto

* git --version -> Muestra la versión de git.
* git config -h -> Opciones del comando config
* git config –l -> Configuración del repositorio.
* git config --global user.name “Samuel Castro Domingue”.
* git config --global user.email [fenixscd@gmail.com](mailto:fenixscd@gmail.com).
* git config user.name “Samuel Castro Domingue”.
* git config user.email [fenixscd@gmail.com](mailto:fenixscd@gmail.com).
* git config --global alias.co checkout
* git config --global alias.ci commit
* git config --global alias.st status
* git config --global alias.unstage 'reset HEAD --'
* git config --global alias.visual '!<comando externo a git>'

**Comandos**

Las opciones de los comandos de una sola letra suele ser solo un guion “-”, las que son una palabra dos “--”.

* **git init**

Crea un repositorio.

* **git status**

Estado actual de los ficheros.

* **git add <Nombre del archivo o expresión regular>**

Para añadir los ficheros que no se están siguiendo (Untracked). Git soporta expresiones regulares y si ponemos **“.”** Nos añadirá todos los archivos que estén Untracked, si ponemos **git add \*.txt** añade todos los archivos que tengan extensión txt

* **git commit**

Añade al LOCAL REPOSITORY.

**< Nombre del archivo o expresión regular >**

**-m “<Comentario>”**

**-a** Los archivos que ya están trackeados podemos commitear sin pasar por add.

**--amend** Añade los cambios al último commit, abrirá el comentario por si queremos modificarlo.

Ejemplos:

git commit -am “comentario”

git commit -a --amend -> Abrirá para modificar el comentario

* **git log**

Para ver todos los commits que se han producido, se lee de abajo a arriba.

**-- git log --online**

Saca los commits en una sola línea.

**--online <Numero commits>**

Muestra el número de commits indicados, con el nombre relativo si ponemos 5 solo imprime los últimos 5.

**--<archivo>** Muestra en los commits que está involucrado un archivo.

**--grep=’expresión regular’** Muestra los commits en los que intervienen archivos que cumplen la expresión regular.

**--no-merges** Eliminar del listado los que sean commits de merges

* **git show**

Muestra la información de los ficheros que hay dentro del commit.

* **git show-branch**

Muestra los commits

**--more=n** Muestra n commits

* **git rm --cached < Nombre archivo o expresión regular >**

Devuelve el fichero al WORKING DIRECTORY siempre que no se halla hecho ningún commit

* **git reset HEAD < Nombre archivo o expresión regular >**

Devuelve el fichero al WORKING DIRECTORY después de haber sido commiteado.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_

| Working Área | | Stagin Área | |

| | git add <file> → |Index | |

| |← git rm –cached <file>| | |

| | | | git commit<file> → |

| |← git reset HEAD <file>| | |

|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_| |\_

* **git checkaut**

**<Nombre de la rama>** Nos mueve a la rama especificada.

Si añadimos un indicador nos mueve a ese commit.

**^** Tantos como para atrás queramos ir.

**~<Numero>** Se número de commits que queremos retroceder.

**<Id de un commit>** Nos mueve a ese commit como mínimo los 7 primeros dígitos

**--<Archivo o expresión regular>** Como último parámetro se le pude indicar a que archivos afecta, como único parámetro, recupera si hemos borrado o modificado algún archivo.

* **git branch**

Muestra las ramas que tenemos.

**<Nombre de la rama>** No posiciona en la rama indicada.

**-b <Nombre>** Crea una rama con el nombre especificado y no mueve a la rama creada.

* **git reset**

Eliminar

**HEAD <archivo o expresión regular>** Devuelve el archivo del Index al Working Área

**--soft <Id Commit>** Devuelve el archivo del Local Repositori al Index. Git reset –soft HEAD^

**--mixed <Id Commit>** Devuelve el archivo del Local Repositori al Working Área. Git reset –mixed HEAD^

**--hard<Id Commit>** Elimina todos los commits hasta llegar al especificado incluyendo los archivos y cambios llegar a borrarlo.

**reset --hard HEAD~1** Elimina el ultimo commit.

* **remote**

**-v** Lista todos los repositorios remotos que tenemos.

**rm <Nombre del remoto>** Elimina el repositorio remoto

**add<Nombre del repositorio><Ruta del repositorio>** Añade repositorio.

* **pull**

Descargar los cambios al repositorio remoto.

all

* **push<remote><rama>**

Subir los cambios del repositorio remoto.

**-f <remote><rama>**

* **fetch<remote>**

Recupera todo los datos del remoto.

.**gitignore**

La el archivo de configuración global del gitignore el .gitignore\_global se encuentra en el directorio del usuario.

Luego hay un git ignore para cada proyecto.

# ->Comentarios.

¡ -> Niega una acción.

\* -> Cualquier cosa

**Configuración del ssh para Windows**

En la consola de git

La carpeta de claves ssh está en la carpeta del usuario .ssh

Ejecutamos el comando ssh-keygen

Nos pedirá un nombre, por defecto viene id\_rsa y una contraseña, eso genera 2 archivos con el nombre indicado, uno que no tendrá extensión, que es la clave privada, y otro con extensión .pub que es la clave pública.

Después de generar las claves vamos al archivo nombre.pub, lo abrimos y copiamos el contenido, vamos a nuestro gestor de repositorios remotos (GitHub, Bibucket,…), y en el apartado de configuración, nos posicionamos en el apartado de claves SSH, añadimos una nueva clave y pegamos el contenido del archivo nombre.pub.

Tenemos que iniciar el agente de ssh “eval $(ssh-agent -s)” ahora ya podemos añadir a nuestro agente la clave privada.

ssh-add ~/.ssh/nombre, ya estaría funcionando.

**Recursos**

**Erramientas**

* SourceTree
* SmartGit/Hg

…or create a new repository on the command line

echo "# proyecto\_instituto" >> README.md

git init

git add README.md

git commit -m "first commit"

git remote add origin https://github.com/fenixscd/proyecto\_instituto.git

git push -u origin master

…or push an existing repository from the command line

git remote add origin https://github.com/fenixscd/proyecto\_instituto.git

git push -u origin master