# COMANDOS GIT

* **git help <command>**
* **git clone <uri> namedir***# clona usando como nombre de directorio namedir.*
* **git add <dir>***# añade recursivamente todos los archivos del dir.*
* **git diff --staged***#compares staged changes with last commit*
* **git commit -v***# muestra el diff en el editor*
* **git commit -a -m ”***#automatically stage tracked files. No hace falta git add*
* **git rm --cached <file or regexp>***#Git no realiza un seguimiento del archivo, pero los deja en el directorio de trabajo. Útil cuando se olvida añadir archivos al .gitignore y ya hemos agregado dichos archivos al repositorio.*
* **git rm <file>***#borrarlos con git siempre.*
* **git rm -f <file>***# si ya está modificado y en el index.*
* git mv <file> <renamed\_file>
* **gitk***# tcl/tk. Herramienta gráfica para git*
* **git commit --amend***#Modificar el mensaje del último commit*
* **git reset HEAD <file>***# to unstage*
* **git checkout -- <file>***# Descartar cambios en el directorio de trabajo.*

**AÑADIR ARCHIVOS**

* **git add -i***#interactive staggin*
* **git add -p***#crea patch*

**AÑADIR VARIOS REPOSITORIOS REMOTOS**

* **git remote add bitbucket git@bitbucket.org:algui91/grado\_informatica\_tsi\_practicas.git** *# Añadir un nuevo repositorio remoto con el nombre deseado. Por ejemplo si ya tenemos uno en github y queremos añadir otro para bitbucket*
* **git push -u bitbucket –all** *# Subir el proyecto a bitbucket. A partir de ahora se puede seleccionar a qué repo publicar con***git push nombre\_repo\_remoto**

**CONFIGURATION**

* + **git config --global <opcion> <valor>***#global para usuario, system todos y sin nada, especifico para el repo.*
  + **git config {key}***# muestra el valor de key*
  + **git config --global core.editor <editor>***#cambia el editor por defecto*
  + **git config --global commit.template $HOME/.gitmessage.txt***#plantilla para commits*
  + **git config --global core.pager ‘more|less’***#paginador por defecto, puedes usar cualquiera*
  + **git config --global user.signingkey <gpg-key-id>***# clave gpg para firmar tags*
  + **git config --global core.excludesfile <file>***#como gitignore*
  + **git config --global help.autocorrect 1***# autocorrige cuando se escribe un comando incorrecto. Solo en git >= 1.6.1*
  + **git config --global color.ui true***# colorea la salida de git. Valores: true|false|always*
  + **git config --global core.autocrlf input***#para que usuarios linux no tengan problemas con los retornos de carro de windows*
  + **git config --global core.autocrlf true***#para usuarios de windows*
  + **git config --global core.whitespace trailing-space, space-before-tab, indent-with-non-tab, cr-at-eol***# respectivamente: busca espacios al final de línea, busca espacios al inicio de tabulación, busca líneas con 8 o más espacios en lugar de tabulaciones, acepta retornos de carro*
  + **git apply --whitespace=warn <patch>***# advierte de errores de espacios antes de aplicar el patch. Con --whitespace=fix intenta arreglarlos*

**BRANCH**

# las ramas simplememte son punteros a distintos snapshots

* + **git branch <nombre-rama>***#crea rama. Puntero al commit actual*
  + **git checkout <nombre-rama>***#cambiar a la rama especificada.*
  + **git checkout -b <nombre-rama>***#crea y cambia de rama*
  + **git merge <rama>***# Mezcla la rama actual con <rama>*
  + **git branch -d <rama>***#elimina la rama*
  + **git push origin --delete <branchName>***# Elimina una rama del servidor*
  + **git mergetool***#Herramienta gráfica para resolver conflictos*
  + **git branch***# lista ramas*
  + **git branch -v***# lista ramas mostrando último commit*
  + **git branch --merged***#lista ramas que han sido mezcladas con la actual. Si no tienen un \*, pueden borrarse, ya que significa que se han incorporado los cambios en la rama actual.*
  + **git branch --no-merged***#lista ramas que no han sido incorporadas a la actual.*