Ver mas tarde:

<http://bloonde.com/blog/git-presentacion/>

<https://git-scm.com/book/es/v1/Fundamentos-de-Git-Guardando-cambios-en-el-repositorio>

<https://blogdopata.github.io/Taller_Github/#/>

<https://eltanquedeideas.wordpress.com/2015/12/22/el-flujo-basico-de-git-en-7-pasos/>

|  |  |
| --- | --- |
| Mkdir myrepo..  Git init | Inizializa un directorio de git vacio, SIN crea los elementos por defecto de git |
| Mkdir myrepo..  Git init –bare | Inizializa un directorio de git vacio, además crea los elementos por defecto de git, Los archivos: Config, description, HEAD, hooks, info, objects, refs |
| Git add . | Agrega todos los archivos al “area de preparacion”, todos, sin importar si se este haciendo seguimiento o “tracked”. |
| Git add –A | se agregan todos los archivos del “directorio de trabajo” a “área de preparacion”, y que se estean haciendo seguimiento o “tracked” |
| Git add myfile.txt |  |
| Git comit –m “mensaje de comit” |  |
| Git commit –amend | Cambiar el mensaje del ultimo commit |
| Git config –global user.email “martin@gmail.com” |  |
| Git config –global user.name “martin omar” |  |
| Git config –global core.editor subl | Asignar el editor de texto por defecto *ejemplo “subl” para “sublime text”, en el caso de sublime, en Windows, es necesario agregarlo como “VARIABLE DE ENTORNO” #1.en la sección “variables de sistema”, #2.en la fila “PATH”, editar.  #3.Agregar al final punto y coma (;) y la dirección de la CARPETA de sublime text,  ejemplo:. “C:\Program Files\Sublime Text 3” #4.luego TESTEAR de la ventana del cmd y escribir “subl” si se ejecuta, se asignó correctamente.* |
| Git diff | Ver la diferencia de contenido, muestra los cambios entre commits |
| Git diff –staged | Ver la comparativa de líneas de contenido de un archivo, staged, para indicar la comparativa con el estado “área de preparacion” |
| Git list |  |
| Git config –list | Ver el listado de configuracion |
| Git log | Ver el historial de los commits realizados, muestra todos los commits empezando con el mas reciente |
| Git log –oneline –decorate –graph | Ver de manera formateada los comits |
| Git status | Ver el estado actual de los seguimiento de cambios |
| Git stash ¿?? | guarda todos los cambios y estados en un directorio alterno y regresa el directorio de trabajo al estado del HEAD |
| Git help | Muestra las opciones de ayuda |
| git remote add origin https://github.com/martinomar/MyPersonalRepo | Agrega el repositorio de origen remote |
| Git push –u origin master | Guarda los cambios pendientes de la rama master, en “origin” |

Fuentes:<https://eltanquedeideas.wordpress.com/2015/12/22/el-flujo-basico-de-git-en-7-pasos/>

Terminología - Traducciones

|  |  |
| --- | --- |
| to unstage. | para dejar de escena, etapa, escenario, escena, teatro, grado, plataforma |
| detached. | separado   |  |  |  | | --- | --- | --- | | *adjetivo* | | | |  | separado | separate, asunder, detached | |  | suelto | loose, free, single, detached, separate, flyaway | |  | desprendido | detached, disinterested | |  | objetivo | objective, factual, realistic, detached | |  | desinteresado | disinterested, selfless, uninterested, indifferent, cheerless, detached | |  | postizo | false, artificial, dummy, detached | |
| Stage | Escenario   |  |  |  |  | | --- | --- | --- | --- | | sustantivo | | | | |  | la | etapa | stage, phase, lap, degree | |  | el | grado | degree, grade, extent, rank, stage, point | |  | el | escenario | stage, scene, setting | |  | la | escena | scene, stage, setting | |  | el | teatro | theater, stage, playhouse, theatre, play-acting | |  | la | plataforma | platform, stage, bench | |  | el | escalón | step, stair, stage, echelon, rung, ladder | |  | el | estrado | dais, stage, platform, podium, bandstand | |  | el | portaobjeto | stage | | verbo | | | | |  | organizar | | organize, arrange, host, be organized, stage, put together | |  | escenificar | | stage | |  | efectuar | | make, effect, effectuate, bring about, stage, prosecute | |  | representar | | represent, depict, play, portray, act, stage | |  | arreglar | | fix, arrange, settle, sort, fix up, mend | |
| Detached | Separado   |  |  |  | | --- | --- | --- | | adjetivo | | | |  | separado | separate, asunder, detached | |  | suelto | loose, free, single, detached, separate, flyaway | |  | desprendido | detached, disinterested | |  | objetivo | objective, factual, realistic, detached | |  | desinteresado | disinterested, selfless, uninterested, indifferent, cheerless, detached | |  | postizo | false, artificial, dummy, detached | |

Lista Comandos Git

Susanna Allés Torrent edited this page on 1 May 2016 ·

Lista de comandos útiles para Git

###Para clonar el repositorio de GitHub (sólo una vez):

* git clone https://github.com/susannalles/MinimalEditions.git

###Para subir nuevos materiales a GitHub:

* git init: inicia git al interno de la carpeta
* git add nombre\_archivo.txt: añade el documento (o carpeta) en el area de espera ("stage")
* git commit -m "mi primer mensaje de cambios": describe los cambios realizados
* git remote add origin https://github.com/susannalles/MinimalEditions.git: apunta a la dirección donde deseáis subir el nuevo material
* git push -u origin master": subís los cambios al repositorio remote en GitHub por primera vez

###Push & Pull

* git add \*: añade el documento (o carpeta) en el area de espera ("stage")
* git commit -m "mensaje con los detalles del cambio": describe los cambios realizados
* git push origin master: subís los cambios a GitHub
* \*git push origin [branch]: subís los cambios al repositorio remote en GitHub. Asegurar de escribir el nombre del branch que quieres subir sus cambios y **nunca subes al master** sin que todos revisamos sus cambios.
* git push origin nombre\_de\_la\_rama, Para enviar los cambios en el HEAD del repositorio local al remoto ejecutamos el comando:

###Sincronizar nuestra copia con el original:

* git pull: baja los cambios del repositorio remoteo a nuestra copia en local

###La Brújula:

* git status: señala lo que se ha modificado en la carpeta de trabajo

###Branches

* git branch: Para ver en que branch estas trabajando.
* git branch [name]: Para crear un branch nuevo. Asegurar de usar git checkout para camibar al branch.
* git checkout [branch]: Para cambiar de un branch a otro. Así por ejemplo, si estamos en "master" y queremos cambiarnos a un branch llamado "classwork", haremos 'git checkout classwork'.
* git branch -d [name]: Para quitar un branch. Es posible que git te da un error. Git no te permite quitar un branch que tiene commits no escritos al origin con esta orden. Si estas *completamente seguro* de que quieres quitar el branch, puedes usar la orden git branch -D [name].
* git checkout -b nombre\_branch: para bajar el contenido de un branch
* git pull origin nombre\_branch: para actualizar el contenido de un branch (y empezar a trabajar en ese branch).
* git checkout -b nombre\_branch: para descargar el contenido de un branch
* git checkout -- file.html: para ignorar los cambios hechos en local (cuando queremos hacer un push)

###Tutoriales & Recursos:

* [Pro Git](http://git-scm.com/book/en/v2) by Scott Chacon and Ben Straub

Fuente:

<https://github.com/susannalles/MinimalEditions/wiki/Lista-Comandos-Git>

commandos----