COMANDOS GIT

|  |  |  |
| --- | --- | --- |
| Comando | Descripción | Ejemplo |
| CONFIGURACIONES INICIALES | | |
| $ git config --list --show-origin | Ver todas las configuración y de donde provienen |  |
| $ git config --global user.name "John Doe"  $ git config --global user.email johndoe@example.com | Configurar siempre que se instala git, usuario y nombre |  |
| $ git config --global core.editor emacs | Configurar el editor de texto, además el *wait*, debe esperar que cierre el editor, antes de continuar con el siguiente paso | git config --global core.editor "code --wait" |
| $ git config --global init.defaultBranch main | Establecer *main* como rama principal |  |
| $git config --list | Verificar configuración |  |
| PEDIR AYUDA | | |
| $git add -h | Mostrar comandos de ayuda, con respecto al parámetro |  |
| INICIAR UN REPOSITORIO LOCAL EN EL EQUIPO | | |
| $ git init | Iniciar git |  |
| $git add \* | Preparar al staging |  |
| $git commit –m “mensaje” | Comitear |  |
| $git clone $ git clone https://github.com/libgit2/libgit2 | Clonar un repositorio remoto |  |
| $ git clone https://github.com/libgit2/libgit2 mylibgit | Crear un repositorio remoto con un nombre diferente del directorio |  |
| $git status -s | Conocer el estado del directorio –s, forma breve |  |
| $ git add README | Para comenzar a rastrear el archivo y preparado para ser comiteado, después de ser modificado |  |
| ESTADOS DE LOS ARCHIVOS | | |
| Hay 2 columnas, la de la izquierda es el estado del área de preparación y la de la derecha indica el estado del árbol de trabajo | | |
| ?? | Nuevos | No se rastrean |
| A | Nuevos que se han agregado al área de preparacion |  |
| M | Son modificados |  |
| GIT IGNORE | | |
| ejemplos |  |  |
| GIT DIFF | | |
| Muestra las líneas exactas agregadas y modificadas | | |
| git diff | Ver lo que se ha cambiado, pero no se ha preparado, lo de color verde es lo nuevo |  |
| Git diff --staged | Ver lo que se ha cambiado, pero en el área de preparación, lo del signo + es lo nuevo, el signo – es lo anterior |  |
| COMMIT | | |
| git commit | Abre el editor de texto previamente configurado |  |
| git commit –v | Abre el editor de texto previamente configurado pero mostrando los cambios que van a realizar |  |
| $ git commit -m "Story 182: fix benchmarks for speed" | Se realizar el commit, directamente |  |
| ELIMINAR ARCHIVOS | | |
| $rm archivo.txt | Se elimina el archivo, preparándolo para la eliminación |  |
| $git rm archivo.txt | Se elimina |  |
| $git commit –v | Se realiza el commit para guardar |  |
| $ git rm --cached archivo.txt | Se puede dejar de rastrearlo, pero permanece en el área de preparación **y se debe realizar el commit** |  |
| GIT MV | | |
| $ git mv file\_from file\_to | Cambiar el nombre de un archivo |  |
| GIT LOG | | |
| Este comando muestra los commit realizados, mostrando algunas caracteristicas,  como lo son el hash, el correo y el usuario de quien realiza el commit | | |
| git log -p -2 | muestra los cambios de los ultimos 2 commits, en que han cambiado |  |
| git log --stat | muestra estadisticas abreviadas de cada commit |  |
| git log --stat -2 | muestra estadisticas abreviadas de cada commit de los ultimos 2  imprime debajo de cada entrada de confirmación una lista de archivos modificados,  cuántos archivos se cambiaron y cuántas líneas de esos archivos se agregaron  y eliminaron. |  |
| git log --oneline | Muestra los commit en una sola linea |  |
| FORMAT | | |
| Se puede especificar su propio formato | | |
| $ git log --pretty=format:"%h - %an, %ar : %s" | Numero de hash  Nombre del autor  Fecha del autor  Mensaje | ca82a6d - Scott Chacon, 6 years ago : Change version number  085bb3b - Scott Chacon, 6 years ago : Remove unnecessary test  a11bef0 - Scott Chacon, 6 years ago : Initial commit |
| LISTA DE ALGUNOS FORMATOS | | |
| %H-> Confirmar hash  %h->Hash de confirmación abreviado  %T->Hash de árbol  %t->Hash de árbol abreviado  %P->Hashes de los padres  %p->Hashes de padres abreviados  %an->Nombre del autor  %ae->Correo electrónico del autor  %ad->Fecha del autor (el formato respeta el --date=option)  %ar->Fecha del autor, relativa  %cn->Nombre del colaborador  %ce->Correo electrónico del colaborador  %cd->Fecha del compromiso  %cr->Fecha del committer, relativa  %s->Sujeto | | |
| GRAPH | | |
| Útil en la visibilidad de los commit | | |
| $ git log --pretty=format:"%h %s" --graph |  |  |
| COMANDOS COMUNES LOG | | |
| **-p** -> Muestra el parche introducido con cada confirmación.  **--stat** -> Mostrar estadísticas de los archivos modificados en cada confirmación.  **--shortstat** -> Mostrar sólo las líneas modificadas/inserciones/eliminaciones del --statcomando.  **--name-only** -> Mostrar la lista de archivos modificados después de la información de confirmación.  **--name-status** -> Muestra también la lista de archivos afectados con información agregada/modificada/eliminada.  **--abbrev-commit** -> Mostrar sólo los primeros caracteres de la suma de comprobación SHA-1 en lugar de los 40.  **--relative-date** -> Muestra la fecha en un formato relativo (por ejemplo, “hace 2 semanas”) en lugar de utilizar el formato de fecha completo.  **--graph** -> Muestra un gráfico ASCII de la rama y el historial de fusión junto a la salida del registro.  **--pretty** -> Muestra las confirmaciones en un formato alternativo. Los valores de opción incluyen oneline, short, full, fullery format(donde especificas tu propio formato).  **--oneline** -> Forma abreviada de " **--pretty=oneline --abbrev-commit** usado juntos" | | |
| FILTROS(--sincey --until) desde - hasta | | |
| **$ git log --since=2.weeks** | Muestra los commits realizados hace 2 semanas |  |
| **$git log --since="2024-08-03" --until="2024-08-03 23:59:59"** | Muestra los commits realizados entre las fechas indicadas |  |
| FILTROS(--author --grep) | | |
| **$git log --author="flmartinez@gmail.com"** | Busca por el autor |  |
| **$git log --grep="modi"** | Busca por palabras clave |  |
| FILTROS(--all-match) | | |
| **$git log --grep="Modi" --author="flmartinez" --allmatch** | Realiza la búsqueda de los criterios que coincidan con grep y el autor, gracias  al allmatch(el grep es sensible a mayusculas y minusculas) | |
| FILTROS | | |
| **$ git log -S function** | Es una herramienta muy útil para rastrear la historia de un código específico dentro de un repositorio. | |
| **$ git log -- ruta** | Es importante cuando se quiere conocer los commits que se han hecho en un archivo en específico, indicándole la ruta | |
| LISTA DE COMANDOS | | |
| **-<n>** -> Mostrar sólo las últimas n confirmaciones.  **--since,--after** -> Limite las confirmaciones a aquellas realizadas después de la fecha especificada.  **--until,--before** -> Limite las confirmaciones a aquellas realizadas antes de la fecha especificada.  **--author** -> Mostrar únicamente las confirmaciones en las que la entrada del autor coincida con la cadena especificada.  **--committer** -> Mostrar únicamente confirmaciones en las que la entrada del confirmador coincida con la cadena especificada.  **--grep** -> Mostrar únicamente confirmaciones con un mensaje de confirmación que contenga la cadena.  **-S** -> Mostrar únicamente confirmaciones que agreguen o eliminen código que coincida con la cadena. | | |
| EJEMPLO | | |
|  | | |

Investigar global, system y local

Investigar git difftool