Git Cheatsheet – Carlos Cepeda

git init <directory>	Cree un repositorio de Git vacío en el directorio especificado. Corre sin argumentos para inicializar el directorio actual como un repositorio de git.
git clone <repo>	Clone el repositorio ubicado en <repo> en la máquina local. El repositorio original puede ser ubicado en el sistema de archivos local o en una máquina remota a través de HTTP o SSH
git config user.name <name>	Defina el nombre del autor que se utilizará para todas las confirmaciones en el repositorio actual. Desarrolladores comúnmente usa --global flag para establecer opciones de configuración para el usuario actual.
git config --global credential.username "Nuevo_usuario"	Permite iniciar sesión en git desde la terminal, permite forzar el cambio de usuario en un equipo.
git add <directory>	Organice todos los cambios en <directory> para la próxima confirmación. Reemplace < directory > con un <archivo> para cambiar un archivo específico. Para agregar todos los documentos usar el comando: git add .
git commit -m "<mensaje>"	Confirme la instantánea por etapas, pero en lugar de iniciar un editor de texto, use <mensaje> como mensaje de confirmación.
git status	Enumere qué archivos están almacenados, no organizados y sin seguimiento.
git log	Muestra todo el historial de confirmaciones utilizando el formato predeterminado. Para la personalización, consulte las opciones adicionales.
git diff	Muestre cambios sin etapas entre su índice y directorio de trabajo.
git config --global --edit	Abra el archivo de configuración global en un editor de texto para editarlo manualmente
git config --system core.editor <editor>	Abra el archivo de configuración global en un editor de texto para editarlo manualmente.
git commit --amend	Reemplace la última confirmación con los cambios por etapas y la última confirmación conjunto. Úselo sin nada preparado para editar el mensaje de la última confirmación
git rebase <base>	Vuelva a basar la rama actual en <base>. <base> puede ser un ID de confirmación, nombre de la rama, una etiqueta o una referencia relativa a HEAD.
git reflog	Muestra un registro de cambios en HEAD del repositorio local. Agregue la marca --relative-date para mostrar la información de la fecha o --all para mostrar todas las referencias.
git branch	Enumere todas las ramas de su repositorio. Agregue un argumento <branch> a cree una nueva rama con el nombre <brach>.
git checkout -b <branch>	Cree y consulte una nueva rama llamada <branch>. Agregar -b para pagar una rama existente.

git merge	Fusionar <rama> en la rama actual.
git remote add <name><ulr>	Cree una nueva conexión a un repositorio remoto. Después de agregar un control remoto, puede usar <name> como atajo para <url> en otros comandos.
git fetch <remote> <branch>	Obtiene un <branch> específico, del repositorio. Salir de < branch > para buscar todas las referencias remotas
git pull <remote>	Obtener la copia del control remoto especificado de la rama actual y fusionarlo inmediatamente en la copia local.
git push <remote> <branch>	Empuje la rama a <remote>, junto con las confirmaciones necesarias y objetos. Crea una rama con nombre en el repositorio remoto si no existe.
git revert <commit>	Cree una nueva confirmación que deshaga todos los cambios realizados en <commit>, luego aplíquelo a la rama actual.
git reset <file>	Elimine <archivo> del área de preparación, pero deje el directorio de trabajo sin alterar. Esto desestabiliza un archivo sin sobrescribir ningún cambio.
git clean -n	Muestra qué archivos se eliminarían del directorio de trabajo. Utilice el indicador -f en lugar del indicador -n para ejecutar la limpieza.
git diff HEAD	Mostrar la diferencia entre el directorio de trabajo y la última confirmación
git diff --cached	Mostrar la diferencia entre los cambios por etapas y la última confirmación
git reset	Restablecer el área de preparación para que coincida con la confirmación más reciente, pero deja el directorio de trabajo sin cambios
git reset --hard	Restablezca el área de preparación y el directorio de trabajo para que coincidan con los más recientes confirma y sobrescribe todos los cambios en el directorio de trabajo.
git reset <commit>	Mueva la punta de la rama actual hacia atrás a <commit>, restablezca el área de preparación para que coincida, pero deje el directorio de trabajo solo.
git reset --hard <commit>	Igual que el anterior, pero restablece tanto el área de preparación como el directorio de trabajo a partido. Elimina los cambios no confirmados y todas las confirmaciones después de <confirmar
git rebase -i <remote>	Rebase de forma interactiva la rama actual en <base>. Lanza el editor para ingresar comandos sobre cómo se transferirá cada confirmación a la nueva base.
git pull –rebase <remote>	Obtenga la copia remota de la rama actual y reemplace git en la copia local. Utiliza git rebase en lugar de fusionar para integrar las ramas
git push <remote> --force	Fuerza el empuje de git incluso si da como resultado una combinación que no es de avance rápido. No use la marca --force a menos que esté absolutamente seguro de saber lo que está haciendo.
git push <remote> --all	Empuje todas sus cambios locales al control remoto especificado.