**Comandos GITHUB**

**Git init** = Inicializar repositorios locales o Working directory.

**Git add** = Agregar archivos al Staging área

**Git status** = para ver en que paso van los archivos

**Git commit** = para hacer copia, versión o foto del proyecto

**Git push =** subir el proyecto a n repositorio en la nube

**Git pull** = trae cambios que han hecho otros desarrolladores

**Git clone** = te clona hace copia del proyecto a nuestro pc o entorno de desarrollo

--------------------------------------------------------------------------------------------------------

**Pwd** para ir al directorio local.

**Git init** crea un directorio y ya

**Git status** para ver archivos que estamos trabajando

**Git add** **inicio.html** agrega archivo al staging area

**Git config - -global user.email “correo”** configure correo de usuario

**Git config - -global user.name “nombre”** configure nombre de usuario

**Git commit –m “texto del cambio”** me hace el commit automáticamente

**Git log** ver historial de commit o de versiones locales

**.gitignore** creamos un archivo con esa extencion y todos los archivos o carpetas que relacionesmos aca los podeos ignorar a la hora de hacer commit o push al repositorio web

Si escribimos git status solo aparece el gitignore

**Git add . =** así agregamos todos los archivos de un aves al staging área

**Git add -A =** así agregamos todos los archivos de un aves al staging área

**Git branch** para ver el master y otros usuarios de versionamiento

**Git branch login** crear una versión alternativa llamada login

**Git checkout login** cambiar a la otra versión aquí se pueden ejecutar los add, commit y push y solo se verán los cambios e esta versión

**Git checkout master** para regresar a la versión original, los cambios entre versiones no se pueden ver entre si

**Git remote add origin + link repositorio web de github** para agregar el proyecto local y enlazar este a un repositorio web

**git remote –v** Comprobar el enlace del repositorio web

**git remote remove origin** quita el enlace del repositorio web

**Git push –u origin master** hacer un push o actualización al repositorio web desde la versión master o login

**Git push –f origin master** forzar un push o actualización al repositorio web desde la versión master o login

**git config --global color.ul true** configurar colores en git hub

**git config --global –list** para ver configuraciones generales que hemos hecho en git

**git checkout + código Chad del log** para regresar a una versión anterior

**git checkout master** Para regresar al ultimo commit

**comandos git log**

<https://git-scm.com/book/es/v1/Fundamentos-de-Git-Viendo-el-hist%C3%B3rico-de-confirmaciones>

**git reset –soft** borra commit sin tocar working area

**git reset --mixed** no toca ni working ni tampoco staging area

**git reset --hard** borra todo no deja nada

**git log > commits.txt**  un historial con todos los commits

**git branch –d** borrar nombre de las ramas

**Fusionar git merge + nombre de la rama absorbida** primero se debe ubicar en la rama master

**Issues** dentro de github podemos agregar notas internas al equipo de trabajo

**Milestone** son grupos de issues que aplican para un proyecto , característica o periodo de tiempo

**git commit --amend -m "prueba text"** cambiar nombre del ultimo commit

**git tag -a v0.19 -m "version 19"** definir versions tag anotadas

**git tag -a v0.19** definir versions tag ligeras

**git tag -a v0.19 -m "version 19" + numero de commit** define un tag en especifico

**git push origin v0.1** Se sube el tag de la versión especifica

**git push origin - -tags** Sube todos los tags

**git fetch origin**  Actualizar el usuario oculto con el reporsitorio de github, después de esto actualizamos el master haciendo un git merge origin/master desde la rama master

**git branch –a** ver ramas ocultas

**con q** se sale del git log

HEAD detached from e449237

38dd60c403ac49e85196300d31f05c8922b88961

e44923776440ab18e13a5a81f6282b95d059238c