**Comandos**

***Comandos no GitHub***

1. **echo “texto” > fichero.txt**
   * **Echo >> fichero.txt texto**
2. **cat fichero.txt** -> Ver contenido
3. **ls -la**
4. **git rm --cached fichero** -> Borrar del área de preparación
5. **git mv fichero1**.**txt fichero2.txt** -> Mueve de directorio/Cambia nombre
6. Borrar:
   * **rm archivo**
   * **rm -d directorio** si esta vacío
   * **rm -r directorio** si está lleno

***Configuración***

1. **Git Bash Here** -> En la carpeta que vamos a utilizar en GitHub
2. **git init** -> Iniciar GitHub
3. **git config --global user.email adrian@gmail.com** -> Configurar con nuestro correo asociado en GitHub
4. **git config --global user.name "adrian"** -> Configurar con nuestro nombre
5. **git config user.name**

***Básicos***

1. **git add fichero** -> añadir al área de preparado
   * **git add .** -> añadir todos al área de preparado
2. **git comit -m "texto"** -> Valida los ficheros que están en preparación
3. **git status** -> ver si esta añadido, comit...
   * color **verde** entre medio, entre zona de trabajo y zona de internet
   * color **rojo** esta en el área de trabajo
   * Estados:
     + **working directory** -> Área de trabajo local
       - **git add** -> staging area
     + **staging area** -> Área de preparación, intermedia entre local y repositorio
       - **git commit** -> repositorio
     + **repositorio**
4. **git log** -> información de las versiones creadas, usuario, etc
   * **git log --oneline** -> Se ve la **clave** de los cambios
   * git log --pretty=format:"%h %an %ar - %s"
   * git log --pretty="%h %an %ar - %s"
   * **git show**

***Resetear***

1. **git restore** fichero -> Desechamos los cambios del área working (local)
   1. **git restore .** -> Todos los ficheros
2. **git reset - - hard HEAD** -> combinación de git checkout y git reset. Nos cargamos lo que tengamos en el área staging y working y volvemos al penúltimo commit, deshacemos el último commit.
   1. **git reset --hard clave** -> Deshace el último commit eligiendo un commit anterior
3. **git revert HEAD** -> Deshacer los cambios del último commit. No muestra una pantalla en negro con los cambios

***Visual de cambios***

1. **git diff** -> Cuando modificamos un archivo, se ve la diferencia en comando.
2. **git difftool**-> Cuando modificamos un archivo, se ve la diferencia en gráfico, si configuramos tkdiff sale en ese modo.
   * **git difftool nombreFichero** -> Para ver de un fichero específico
3. **tkdiff** -> Ver los cambios en grafico mejor.
   * Se descarga y se pone: C:\Program Files\Git\usr\bin.
   * Detecta el nuevo verificación de cambios, poniendo: git difftool y luego Y.
4. **git config --global diff.tool tkdiff** -> Para ponerlo por defecto como visualización de cambios.

***Conexión al repositorio***

1. **git remote add origin https://github.com/NombreUsuario(AdrianLozanoMartinez)/NombreRepositorio.git** -> Enlazar con repositorio que previamente hayamos creado
2. **git branch -M main** -> Conectar con el main, rama principal, cambia de master a main porque master ya no existe
3. **git pull –rebase origin main** -> ORDEN: COMMIT -> PULL -> PUSH. Descargamos directorio nuevo creado para enlazar antes de subir, sino da error.

***Subir al repositorio***

**git push -u origin main** -> ORDEN: COMMIT -> PULL -> PUSH. Subir archivos que están en la carpeta, después del commit

***Copia del repositorio***

**git clone https://github.com/NombreUsuario(AdrianLozanoMartinez)/NombreRepositorio.git**

***Ramas***

1. **git branch nombreRama** -> crear rama, pero antes hay que hacer commit sino error
2. **git branch --delete -D** **nombreRama** -> borrar ramas
3. **git branch -v -a** -> Nos muestra las ramas, en verde y \* la que estamos actualmente
4. **git checkout nombreRama**-> Cambia a la rama del nombreRama creado antes
5. **git merge ramaCreado master** -> Unimos ambas ramas, si creamos en la nueva cosas se nos mete en el master al unirlo.

**Desktop**

* **Creamos en un nuevo repositorio** y dentro de él creamos
  + 1. **New repositorio** -> Creamos -> Aparece en documento y dentro github se ve la carpeta.
    2. **Publish repository** -> Para publicar en la web.
    3. Si queremos meter cosas lo metemos en esa carpeta creada.
* **Añadir repositorio local** para añadir un repositorio ya existente y no meterlo en la carpeta de GitHub como en new repositorio.
* **Creamos ramas y usamos**
  1. **current branch** -> **New branch** -> **Push ->** Crear rama
  2. **current branch** -> Elegir rama
  3. Subimos archivo
  4. **merge ->** Unir ramas con la principal (main)