

=====

A) Video 1 [https://youtu.be/ANF1X42\\_ae4](https://youtu.be/ANF1X42_ae4)

=====

1. Git salva instantánea de nuestros ficheros (código + descripción)  
Instantánea-0  
Instantánea-1  
....  
Instantánea-n  
Se puede volver a una instantánea anterior
2. Una aplicación está formada por muchos archivos  
(por ejemplo, si trabajamos con un Framework)  
Necesitamos una instantánea fija de muchos archivos  
(Git toma una instantánea de todos los archivos)
3. Git permite crear ramas  
Un programa o aplicación se puede dividir en varias ramas o flujos de trabajo.  
En una aplicación pueden trabajar varios programadores.  
Así cada programador se centra en uno de los flujos de trabajo o ramas.  
Cada programador hace instantáneas de sus ramas.
4. Git después puede unir o fusionar las diferentes ramas de una aplicación en una única rama.
5. Git, al fusionar las diferentes ramas, puede detectar conflictos  
(ejemplo: dos programadores hacen modificaciones en una misma parte de la aplicación. Git detecta este conflicto y lo intenta solucionar)
6. Git aumenta su potencia al unirse con GitHub.  
Se descarga Git y se instala. Lo importante es la Consola de Git.  
Visual Studio Code es un IDE que está preparado para trabajar con Git.  
Pildoras Informáticas lugar donde están estos vídeos sobre Git.

-----

```
$ git status -s
$ git add .
$ git log --oneline
$ git commit -m "Mejorado Video-1.txt"
$ git push
```

=====

B) Video 2 <https://youtu.be/qk-GWtcdQek>

=====

1. Se verán estos aspectos:  
Crear repositorio  
Agregar al repositorio  
Respaldar en el repositorio
2. Tenemos tres zonas:
  - Directorio de trabajo  
(carpeta donde están nuestros ficheros)
  - Área de ensayo  
(Staging area)
  - Repositorio local  
(lugar donde se guardan las instantáneas)
3. Una primera aproximación a los comandos y su significado  
\$ git init  
Se ejecuta sólo una vez, al comienzo, y así se crea el repositorio local, en donde vamos a hacer un seguimiento de nuestros ficheros.

```
$ git add
    Con esta orden se llevan los archivos del directorio de trabajo
    al área de ensayo
$git commit
    Con esta orden se llevan los archivos del área de ensayo
    al repositorio local
```

4. Creamos una carpeta donde pondremos nuestros ficheros y será el Directorio de trabajo.  
Ahora, en esta carpeta, botón derecho escoger "Git Bash Here" y se abre la consola de Git en esta carpeta.  
Escribimos "git init" y se crea el Repositorio local en esta carpeta.  
(viendo ocultos aparece .git)

5. Indicar lo que va a ser seguido por Git
- ```
$ git status -s
(da el listado de todo lo que hay en el directorio de trabajo;
los ficheros, con ?? delante, es que no se están siguiendo)
```
- Si queremos seguir sólo un fichero:
- ```
$ git add <fichero> (git add caso.txt)
$ git status -s
(muestra el archivo, con A delante; es que está agregado al área de ensayo,
se le está haciendo un seguimiento)
```

6. Para hacer una instantánea
- ```
$ git commit -m "<descripción>"
```
- Si ahora hacemos
- ```
$ git status -s
```
- (No muestra lo que está respaldado, lo que está en el repositorio.  
Sí muestra lo que se está siguiendo, lo que está en el área de ensayo)
- Si ahora hacemos una modificación en un fichero  
Hasta que no hagamos "git add <fichero>", no se agregará  
al área de ensayo

7. Para listar todas las instantáneas que están en el repositorio local
- ```
$ git log --oneline
```
- (listará tantas instantáneas como commit se hayan realizado)

8. Para restaurar una instantánea concreta
- ```
$ git reset --hard <código de la instantánea>
```
- Si ahora hacemos
- ```
$ git log --oneline
```
- (muestra las instantáneas actuales. Si se restaura a  
una instantánea concreta, las instantáneas posteriores se pierden)

```
-----
git init
git add (se añade lo que se quiere hacer el seguimiento)
git commit
git status -s
git add <fichero>
git commit -m "<descripción>"
git config --global user.username "<nombre>"
git config --global user.email "<email>"
git log --oneline
git log --reverse
git reset --hard <código de instantánea>
```

```
=====
C) Video 3 https://youtu.be/0U1qvTJz0L4
=====
```

```
git status -s
```

```

git add .
git log --oneline
git add .
+
git commit
  = git commit -am "<descripción>"
(para cambiar la descripción de de commit)
git commit --amend
(para abrir el vim)
  :i
  esc
  :wq
(como subir a github)
- crear repositorio en github
  - dar nombre, descripción
  - public
- Quick setup (vamos a subir un proyecto existente)
- Para subir la línea:
  git remote .....
  (esta línea se pone en la consola local, y se sube a github)

```

```

=====
D) Video 4  https://youtu.be/axXlYdyDD3I
=====

```

Editar en GitHub  
 (para sincronizar con el local)

commit changes (guarda en el remoto)

Para traer a local - pull  
 (para bajar)  
 \$ git pull (viene lo de remoto a local)  
 (git fech)

Grabar una primera versión del proyecto (V1.0)  
 Con las tags se especifican versiones  
 En el repositorio -> tags (agregar tags)

En local  
 \$git tag 15-09-20V1 -m "Versión 1 del proyecto"  
 \$git log --oneline (aparece el tag)

Para subir el tag - push  
 \$git push --tags

En la carpeta del proyecto (cerrar todo)  
 Simular un desastre (borrar el proyecto en local)  
 Ahora vamos al remoto (GitHub) y en code cogemos la url  
 Ahora en consola local:  
 \$git clone "url" (en este momento ya tenemos el proyecto en local)

To push the current branch and set the remote as upstream, use  
 \$ git push --set-upstream origin master