

INTRO A GIT JCYL 20'



¿Qué es Control de Versiones?

Contenido



Historial completo

Trabajo en equipo



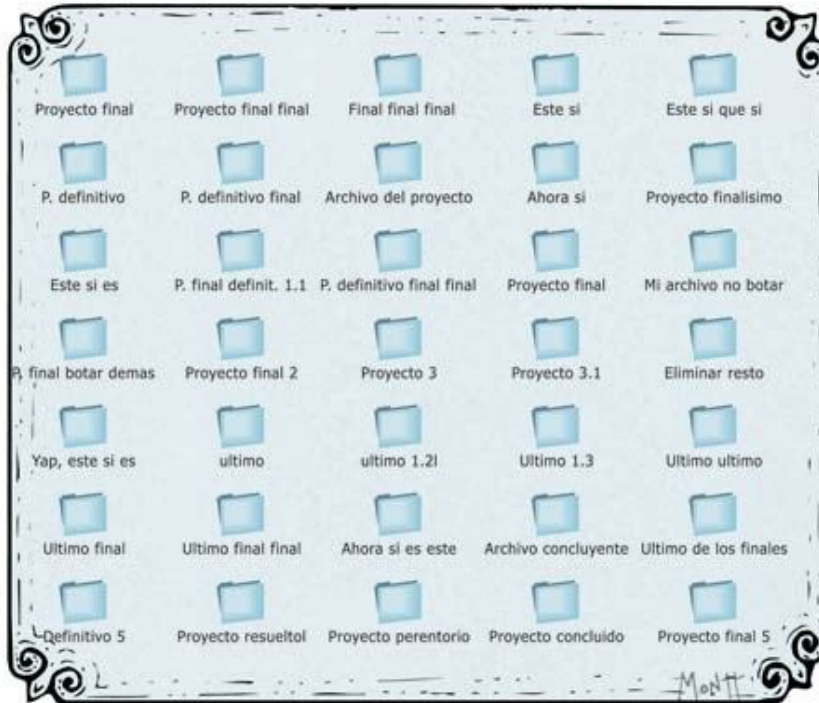
Herramienta colaborativa

Agilidad



Gestión de cambios, tests...

¿Por qué usar control de versiones?



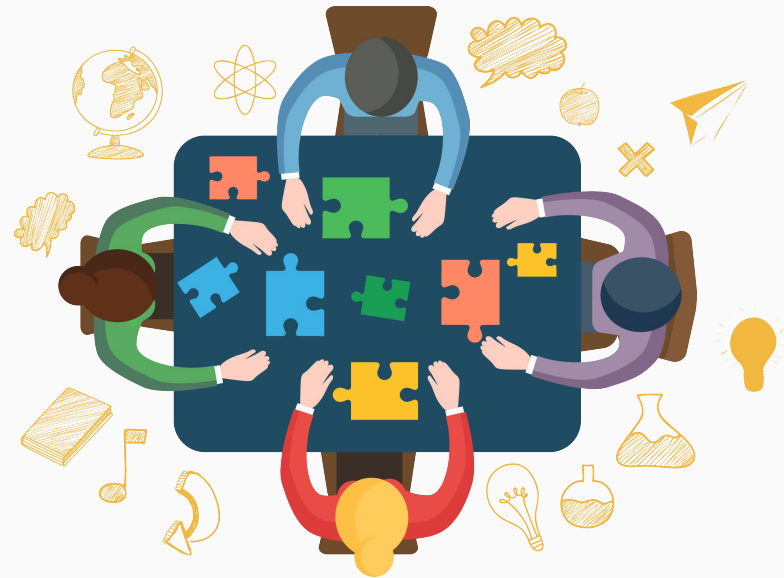
Un proyecto
para
controlarlos
a todos



Registro de
cambios
(deshacer,
rehacer,
comparar..)



¿Por qué usar control de versiones?



Indispensable para trabajo en **equipo**:

- **Método** de trabajo
- **Historial de cambios**
- Trabajar **en paralelo** (en ramas)

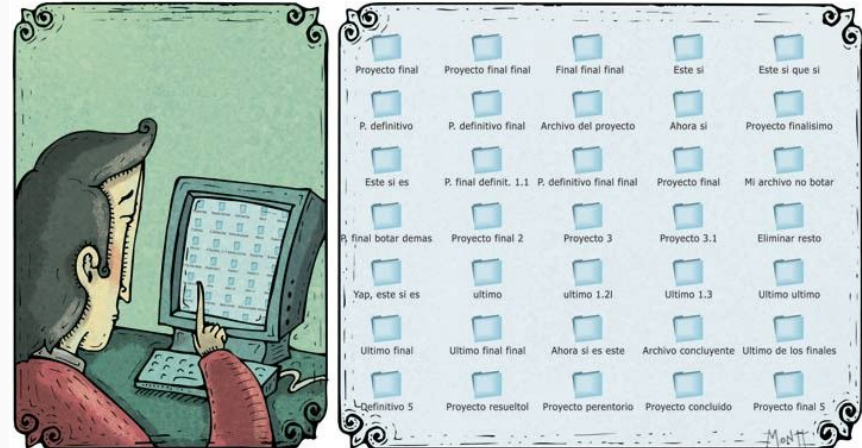


¿Tiene sentido para trabajo individual?

1. Permite trabajar en **diferentes funcionalidades sin mezclarlas**

Por ej: **Cambio de contexto ante un bug crítico**

2. Evito trabajar con muchas carpetas:
Uso **una sola carpeta**



¿Qué es git?

(para 80% de la gente)

ESTO ES GIT. RASTREA EL TRABAJO COLABORATIVO EN PROYECTOS A TRAVÉS DE UN ELEGANTE MODELO DE GRAFOS DISTRIBUIDOS

GUAY. ¿CÓMO SE UTILIZA?

NI IDEA. MEMORIZA ESTOS COMANDOS Y ESCRIBELOS PARA SINCRONIZAR. SI DA ERROR, GUARDA TU TRABAJO EN OTRO SITIO, BORRA EL PROYECTO Y DESCARGA UNA COPIA LIMPIA



¿Qué es git?

(para el que sabe usarlo)



Conceptos Básicos de Git: LOCAL

Las operaciones son **LOCALES**



Offline: No necesitamos conexión para trabajar

Diferencias CVS/SVN

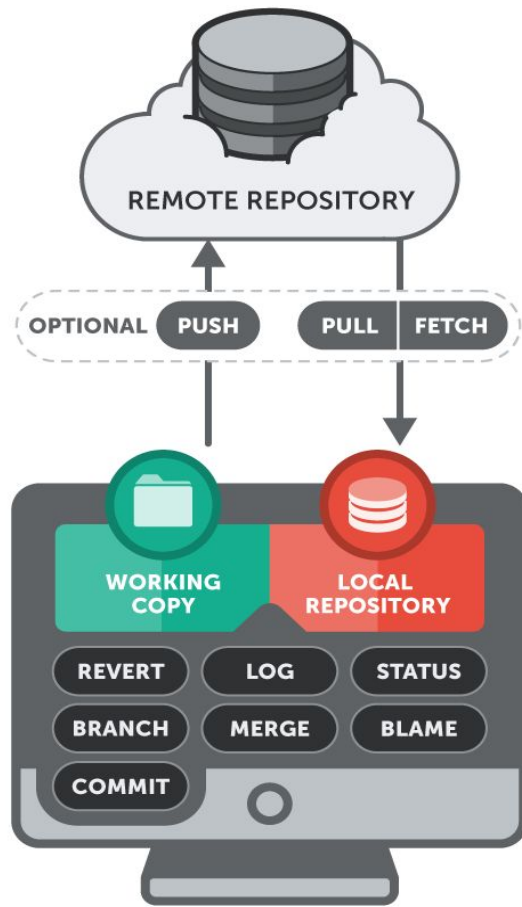
SVN/CVS: todo en servidor

GIT: todo en local

SUBVERSION



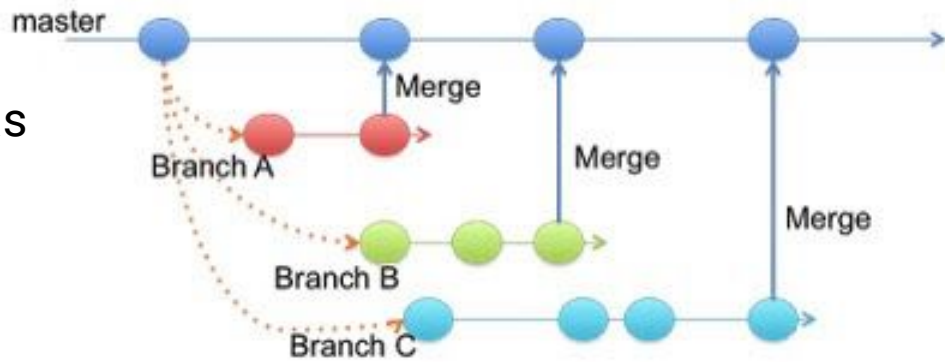
GIT



Ramas en Git

Líneas de trabajo aisladas unas de otras

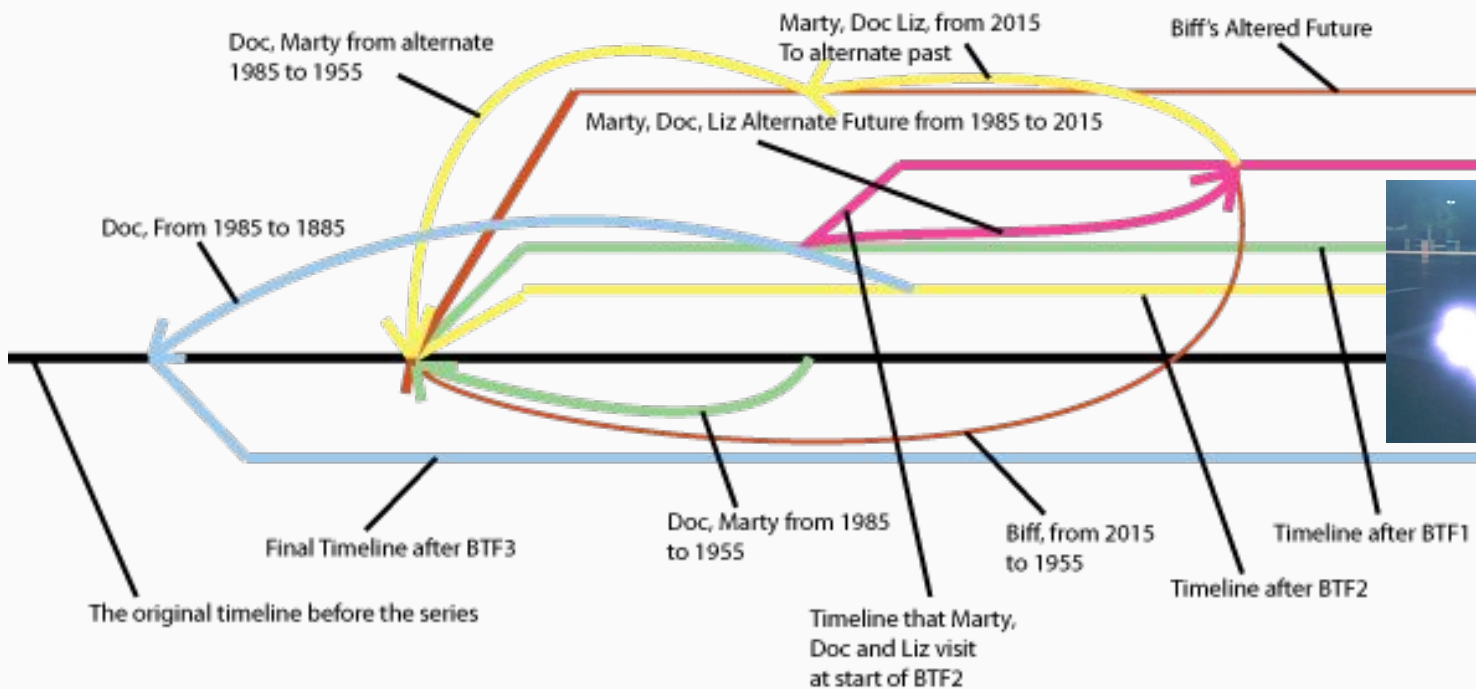
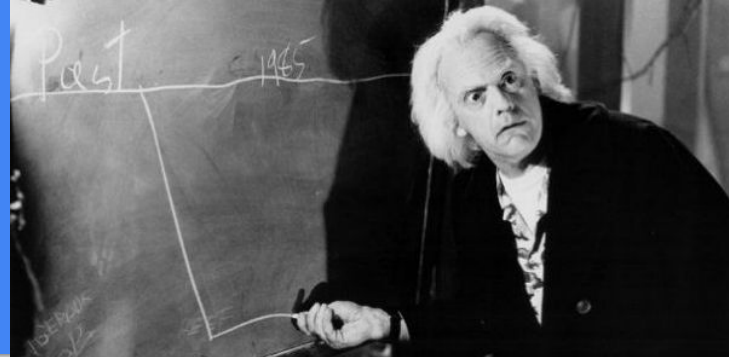
Rama principal por defecto: **MASTER**



Creamos **nuevas ramas** durante el desarrollo (una por cada **funcionalidad**)

Cuando terminamos las fusionamos (**merge**) con la rama principal

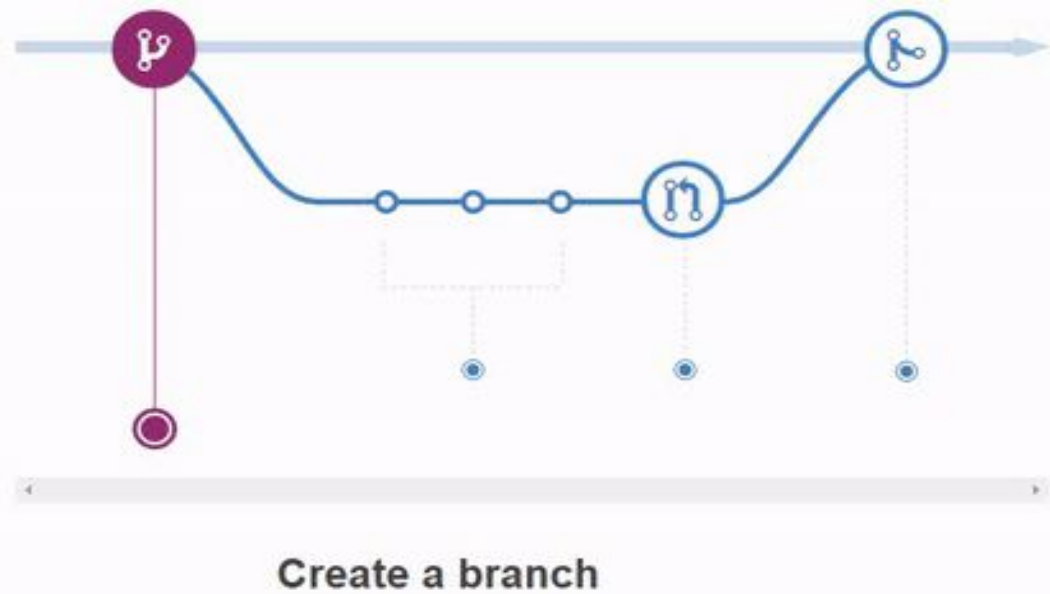
Ramas (Regreso al Futuro)



Trabajo diario

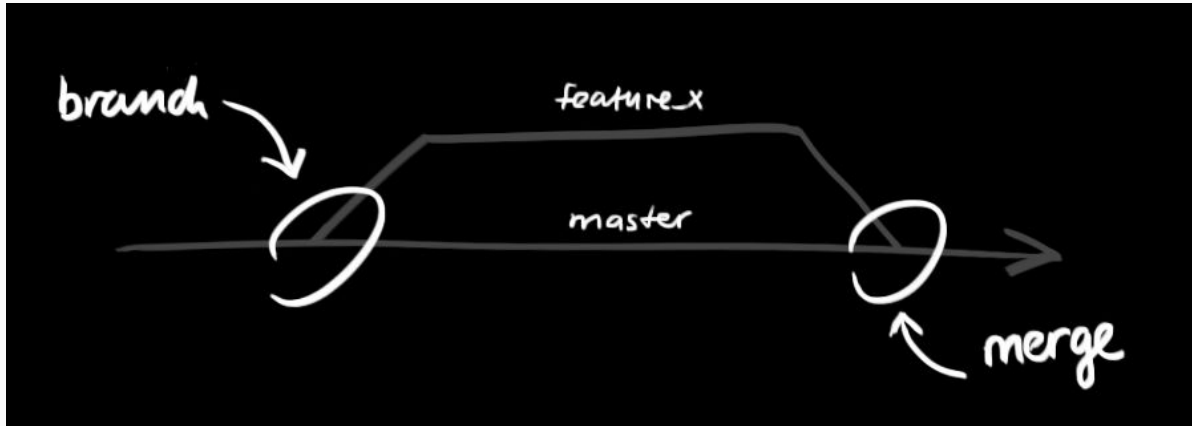
Para cada rama de trabajo

1. Crear una rama y cambiar a ella
2. Hacemos commits de nuestros cambios
3. Cuando acabamos, **Sincronizamos** con el **repositorio remoto**:
 - a. pull → incluir los cambios que hayan subido otros
 - b. push → **subir** nuestra **rama** al repositorio remoto
4. Solicitar **fusionar** nuestros cambios (merge request) en la rama principal



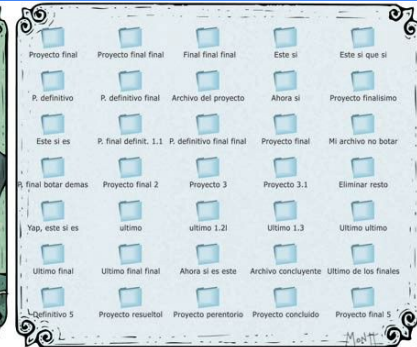
Crear una rama: branch

Una rama = Una funcionalidad

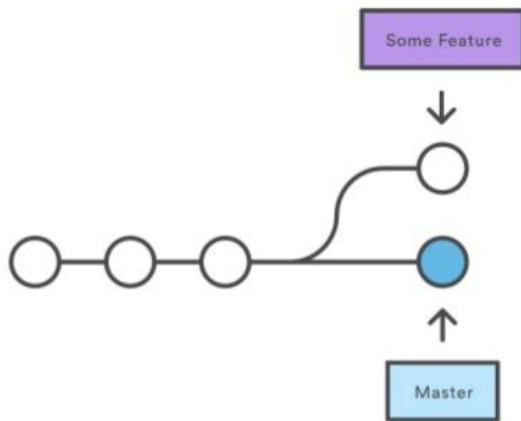


git branch nombre_de_nueva_rama

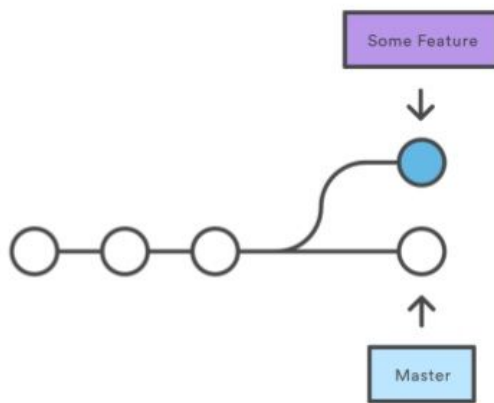
Cambiar de rama: switch / checkout



Para **cambiar a** una rama: **git checkout** rama



git checkout master



git checkout feature



git fetch: Informarse de cambios remotos

git fetch se **entera** de los cambios en remoto



No sobrescribe nuestro directorio de trabajo

Algunos IDEs implementan **autoFetch**



git pull: Bajar y fusionar cambios remotos

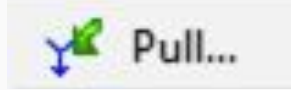
git pull $\Rightarrow$ **git merge** de rama remota indicada

Sobreescribe nuestro directorio de trabajo

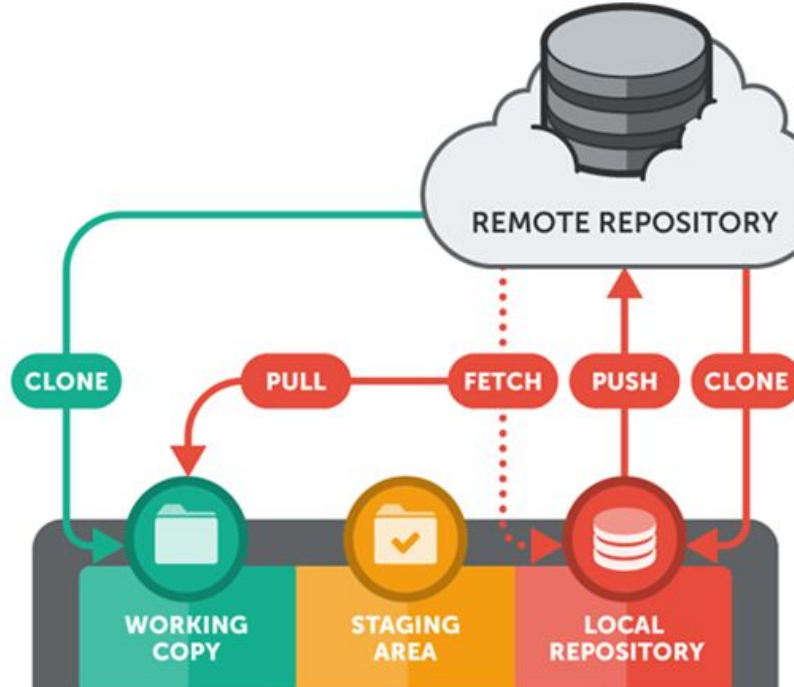


Fetch vs Pull

Pull



Sobrescribe directorio de trabajo
(más agresivo: **pulla**)



Fetch



Se informa de cambios remotos
No sobrescribe

Hacer Push (subir rama a remoto)



PUSH

git push origin rama

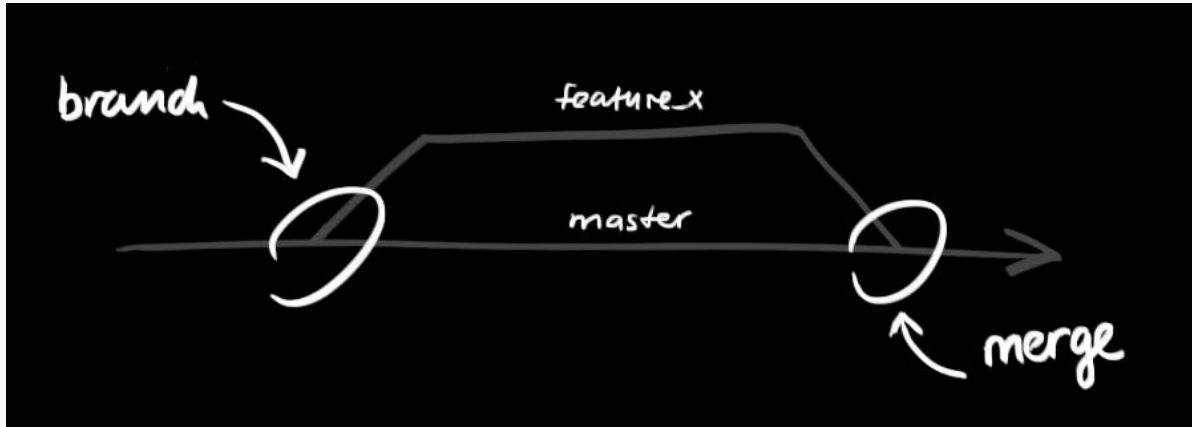
Sube la rama de trabajo (local) al repositorio remoto origin

Si hay conflictos no nos dejará hacerlo

In case of fire 

```
> git commit  
> git push  
> git outofhere
```

Fusionar (merge) una rama



git merge rama

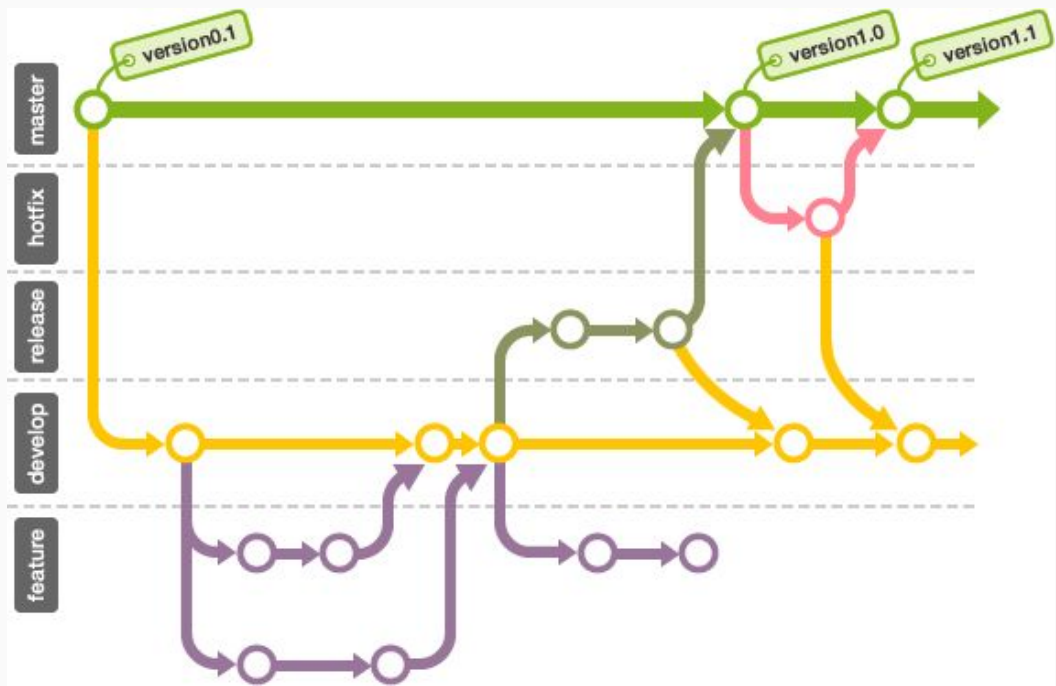
Fusiona la rama indicada en la rama donde estamos trabajando

git tag: Etiquetar versiones



Son **marcas fijas**
(no evolucionan)

Para etiquetar **versiones**



Flujo de trabajo: GitLab Flow

RAMAS estandarizadas:

* **master**: desarrollos parten de ella

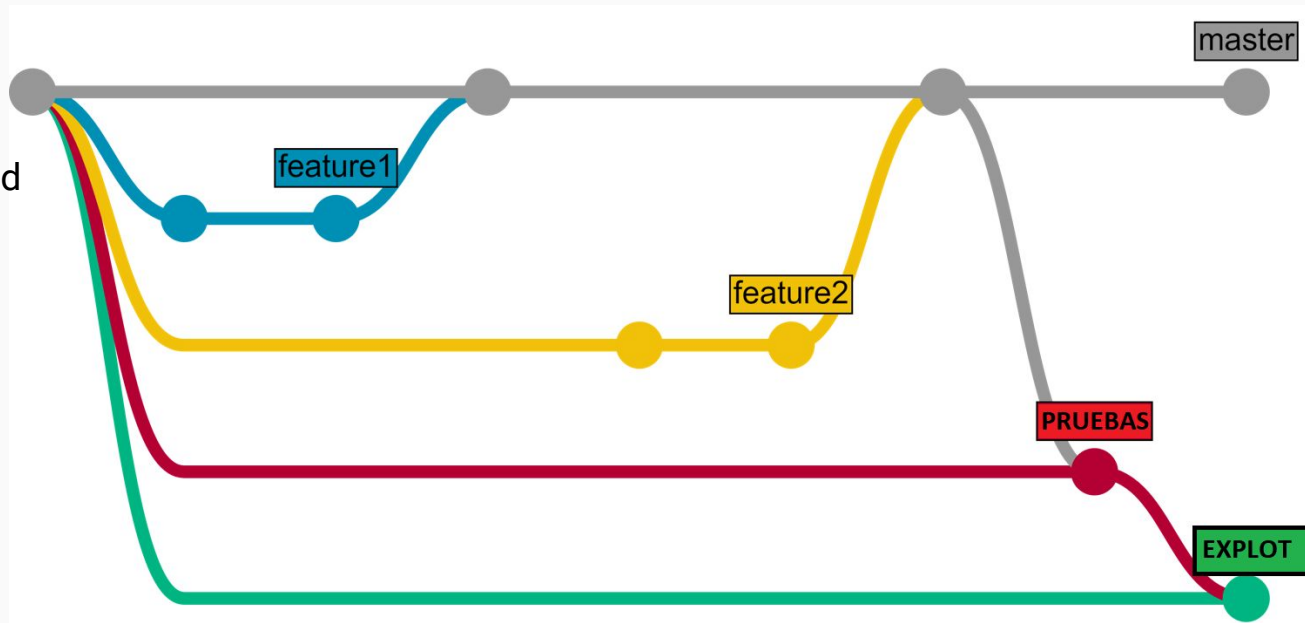
features: una por cada funcionalidad

ejemplos: Nuevo_menu
Corregir_error...

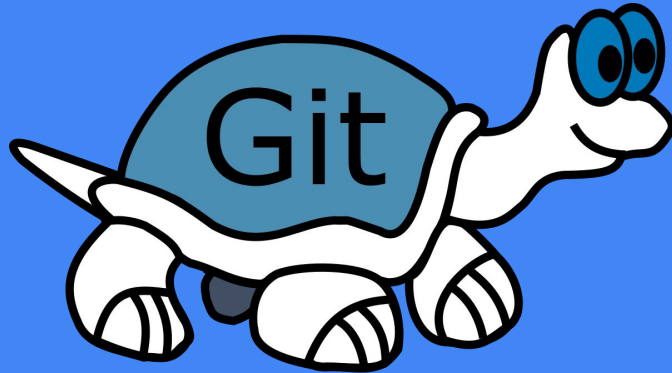
* **rama para cada entorno:**

PRUEBAS

EXPLOTACIÓN



EJEMPLO PRÁCTICO CON TORTOISEGIT

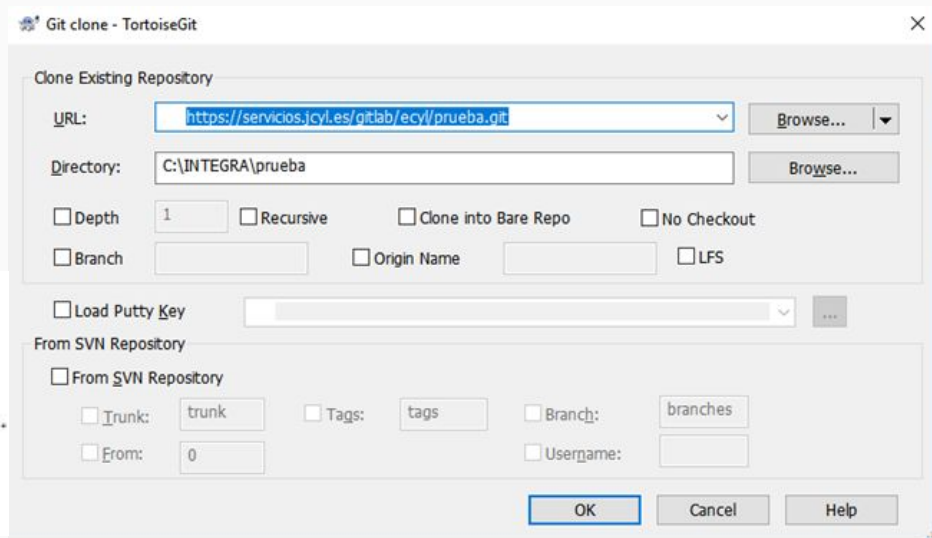
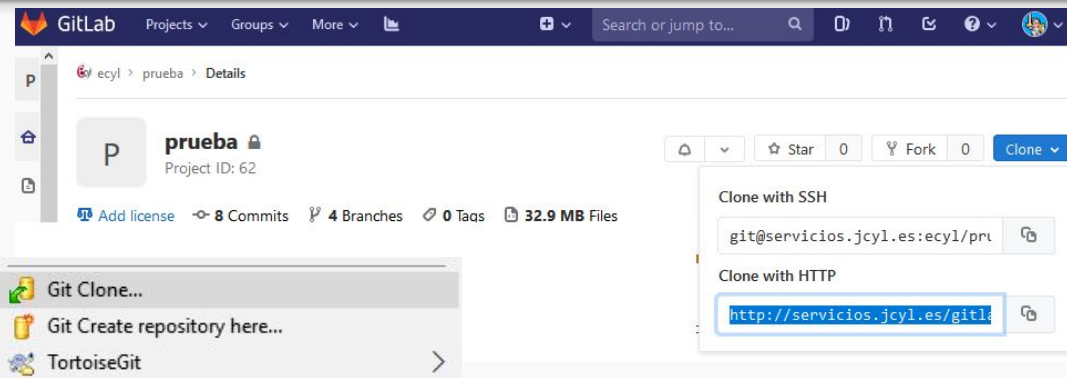


Clone (sólo la primera vez)

1. En **GitLab** copiamos la **url (.git)** del proyecto (**ojo: httpS**)
2. En **Explorador de Windows** botón derecho → **git clone**
3. En **TortoiseGit** pegamos la **url**
Nos pide credenciales (DA o ext)
Si todo va bien dirá **Success**:

```
Cloning into 'C:\INTEGRA\Borrar\prueba'...
POST git-upload-pack (225 bytes)
remote: Enumerating objects: 2165, done.
remote: Counting objects: 100% (2165/2165), done.
remote: Compressing objects: 100% (1284/1284), done.
remote: Total 2165 (delta 790), reused 2165 (delta 790)
Receiving objects: 100% (2165/2165), 32.71 MiB | 10.83 MiB/s, done.
Resolving deltas: 100% (790/790), done.
Checking out files: 100% (1732/1732), done.

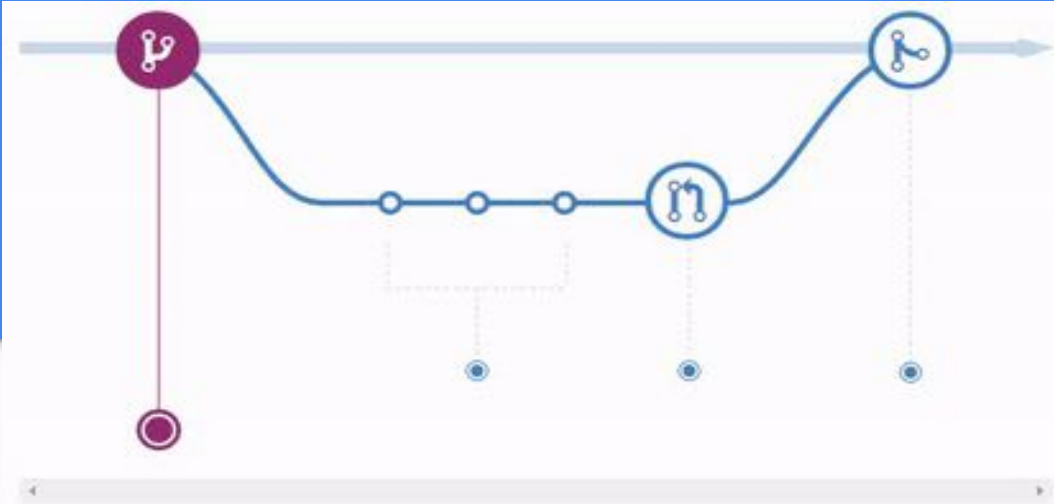
Success (35562 ms @ 29/11/2018 12:54:15)
```



Trabajo diario

Para cada rama de trabajo

1. Crear una rama y cambiar a ella
2. Hacemos commits de nuestros cambios
3. Cuando acabamos, **Sincronizamos** con el **repositorio remoto**:
 - a. pull → incluir los cambios que hayan subido otros
 - b. push → **subir** nuestra **rama** al repositorio remoto
4. Solicitar **fusionar** nuestros cambios (merge request) en la rama principal



Create a branch

Crear nueva rama

1. En **Explorador de Windows**

Botón derecho → **TortoiseGit** → **create Branch**

2. En **TortoiseGit**:

a. Ponemos un **nombre**

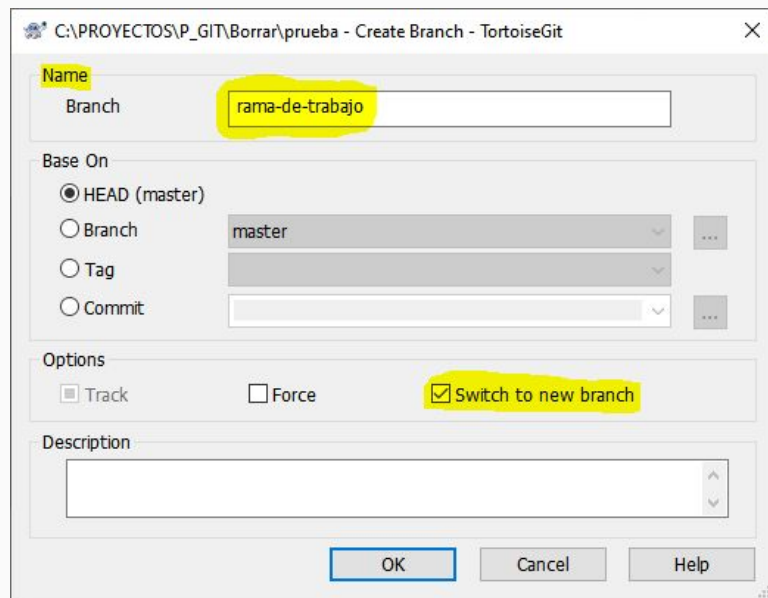
b. Marcamos **“Switch to new branch”**
para trabajar ya sobre la rama

Si todo va bien dirá **Success**

```
git.exe checkout rama-de-trabajo --
```

```
Switched to branch 'rama-de-trabajo'
```

```
Success (1078 ms @ 29/11/2019 10:41:19)
```



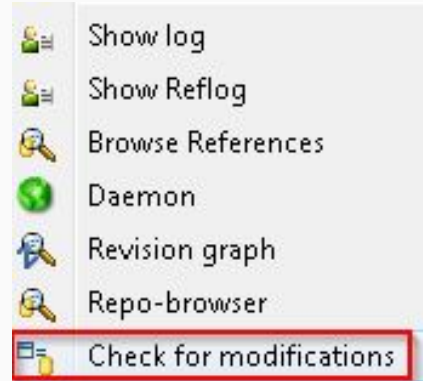
Comprobar modificaciones

Trabajamos en la rama

Git registra todos los cambios

Para ver los cambios que hemos hecho:

Tortoise git → **check for modifications**



C:\PROYECTOS\P_GIT\Borrar\prueba - Working Tree - TortoiseGit						
Path	Extension	Status	Lines added	Lines removed	Last Modified	
Modified Files						
documentos/analisis/VinculacionOfertas.txt	.txt	Modified	1	1	29/11/2019 11:12:26	
documentos/analisis/inscripcion_a_ofertas.txt	.txt	Modified	2	2	29/11/2019 11:12:13	
fuentes/foo.txt	.txt	Missing	0	0		
Not Versioned Files						
Nuevo documento de texto.txt	.txt	Unknown			29/11/2019 11:11:25	

Hacer commits

Para **confirmar los cambios: Git Commit** → **<rama>**

C:\PROYECTOS\P_GIT\Borrar\prueba - Commit - TortoiseGit

Commit to: **rama-de-trabajo** ☐ new branch

Message:

Mensaje corto significativo de la intención de los cambios

1/59

☐ Amend Last Commit

☐ Set author date

☐ Set author

Add Signed-off-by

Changes made (double-click on file for diff):

Check: **All** None Unversioned Versioned Added Deleted Modified Files Submodules

Path	Extension	Status	Lines added	Lines removed
Modified Files				
☒ documentos/analisis/VinculacionOfertas.txt	.txt	Modified	1	
☒ documentos/analisis/inscripcion_a_ofertas.txt	.txt	Modified	2	
☒ fuentes/foo.txt	.txt	Missing	0	
Not Versioned Files				
☐ Nuevo documento de texto.txt	.txt	Unknown		

3 files selected, 4 files total

☒ Show Unversioned Files

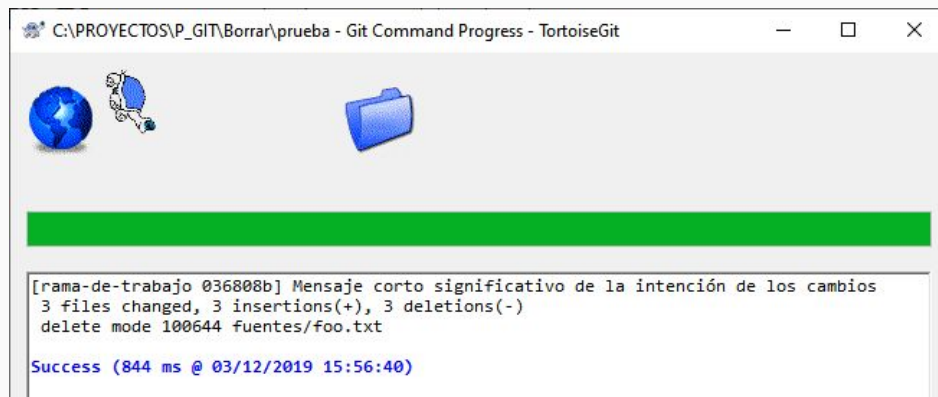
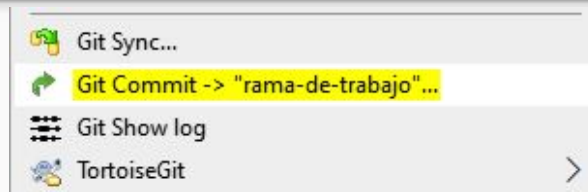
☐ Do not autoselect submodules

☐ Show Whole Project

☐ Message only

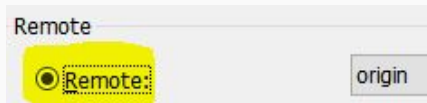
View Patch>>

Commit Cancel Help



Fetch y Pull: Sincronizar con el servidor remoto

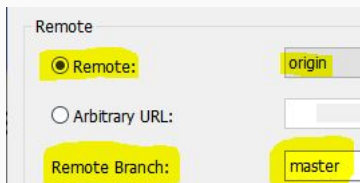
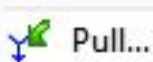
1. **TortoiseGit → Fetch:**
Enterarse de los cambios remotos



```
From https://servicios.jcyl.es/gitlab/ecyl/prueba
= [up to date]      2-error_listado -> origin/2-error_listado
= [up to date]      2-error_listado-2 -> origin/2-error_listado-2
= [up to date]      corregir_error -> origin/corregir_error
7f4fe2e..7d08b50  master -> origin/master

Success (1969 ms @ 04/12/2019 12:25:04)
```

2. **TortoiseGit → Pull:**
Incluir cambios que haya habido en la rama indicada (ej: **master**)



```
From https://servicios.jcyl.es/gitlab/ecyl/prueba
* branch            master -> FETCH_HEAD
= [up to date]      master -> origin/master
Auto-merging nuevo_fichero.java
CONFLICT (content): Merge conflict in nuevo_fichero.java
Automatic merge failed; fix conflicts and then commit the result.
```

git did not exit cleanly (exit code 1) (2000 ms @ 04/12/2019 12:29:17)

Si hay **conflictos**: **TortoiseGit → Check for modifications**: editamos + **commit**

Modified Files				
nuevo_fichero.java	.java	Conflict	5	1 04/12/2019 12:29:16
Not Versioned Files				
Nuevo documento de texto.txt	.txt	Unknown		29/11/2019 11:11:25



Push: Sincronizar con el servidor remoto

3. TortoiseGit → Push: Subir nuestra rama al repositorio remoto



Ref

☐ Push all branches

Local: rama-de-trabajo

Remote:

Destination

☒ Remote: origin

☐ Arbitrary URL:

To <https://servicios.jcyl.es/gitlab/ecyl/prueba.git>
* [new branch] rama-de-trabajo -> rama-de-trabajo
Success (2359 ms @ 04/12/2019 12:53:18)

4. En GitLab pedimos una merge request para que integren nuestra rama de trabajo en master

Project

Repository

Issues

3

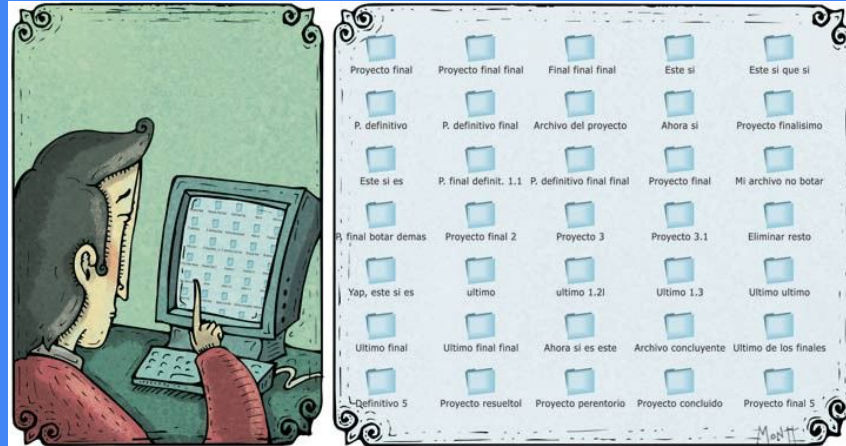
Merge Requests

1

New Merge Request

From rama-de-trabajo into master

Si sólo pudieras quedarte con un par de ideas:



¡Gracias!

