Universidad de Jaén
E.P.S. Linares

Dpto. Ingeniería de Telecomunicación
Área de Ingeniería Telemática

José Manuel Pérez Lorenzo

jmperez@ujaen.es

GITHUB

- El **control de versiones** es un software o sistema que registra los cambios realizados sobre un archivo o conjunto de archivos a lo largo del tiempo. El registro de los cambios permite que se puedan recuperar versiones antiguas o combinar los cambios realizados en los mismos ficheros por personas diferentes.

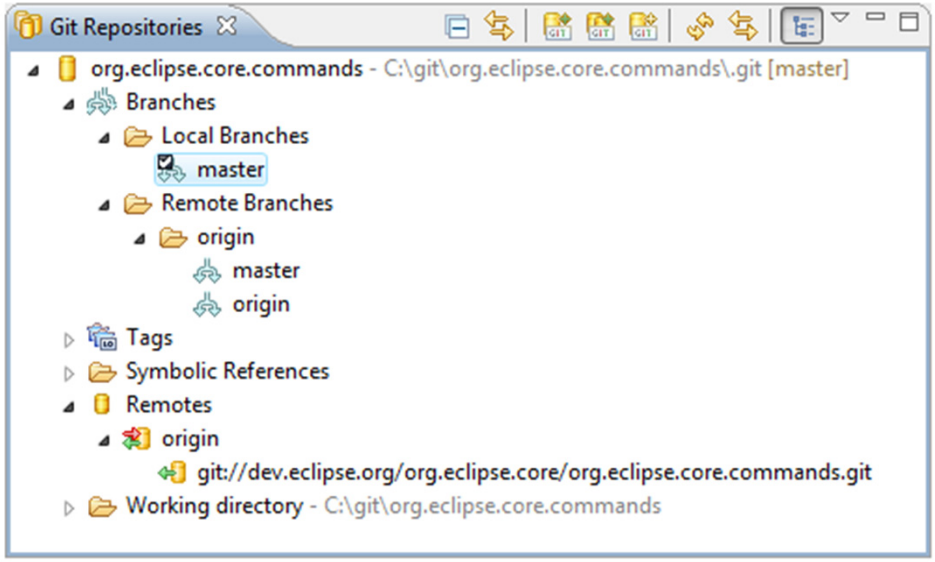
- **Git** es una de las herramientas más usadas para control de versiones. Permite trabajar a cada desarrollador de forma local, en su propio repositorio, a la vez de colaborar en el repositorio común al resto.

- **GitHub** (https://github.com/) es un software de control de versiones que usa Git, permitiendo el uso de repositorios y colaboración.

GITHUB

- Además de tener disponible GitHub via web, se puede acceder al repositorio mediante consola con comandos Git, o mediante otras herramientas específicas.

- Eclipse dispone del plugin Egit, integrando así Git de forma natural en el entorno de programación.

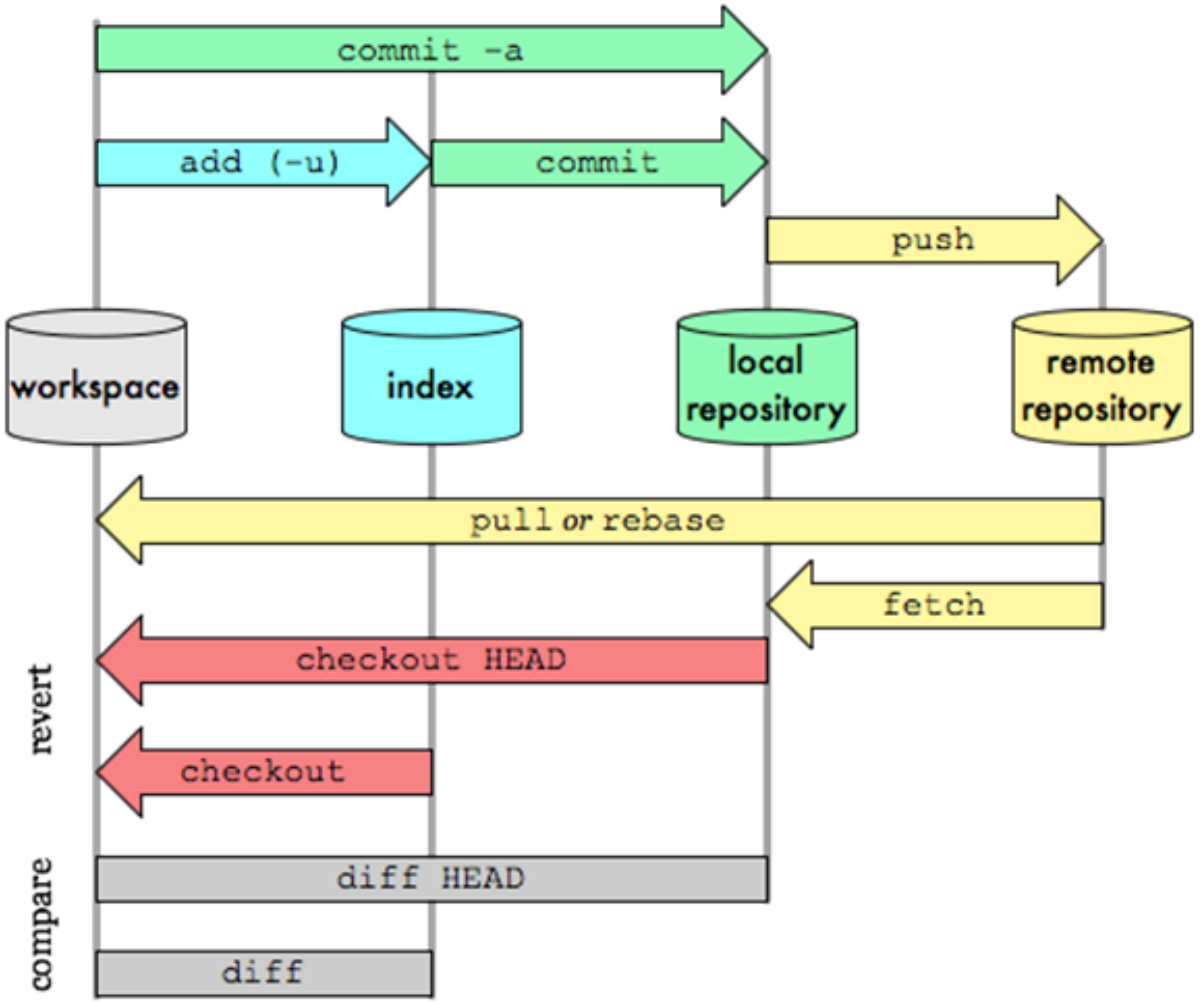


GITHUB

- La integración de Git en Eclipse nos facilita:

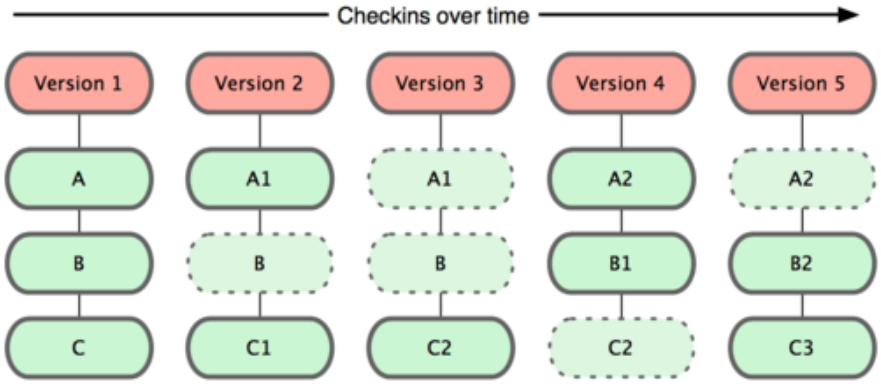
- Revertir un proyecto o archivos a un estado anterior.
- Comparar cambios a lo largo del tiempo.
- Ver quién modificó por última vez alguna parte del proyecto.
- En caso de haber introducido algún error, ver quién lo hizo.
- Recuperar archivos que se hayan dañado o perdido en algún momento.
- etc.

Operaciones Git:

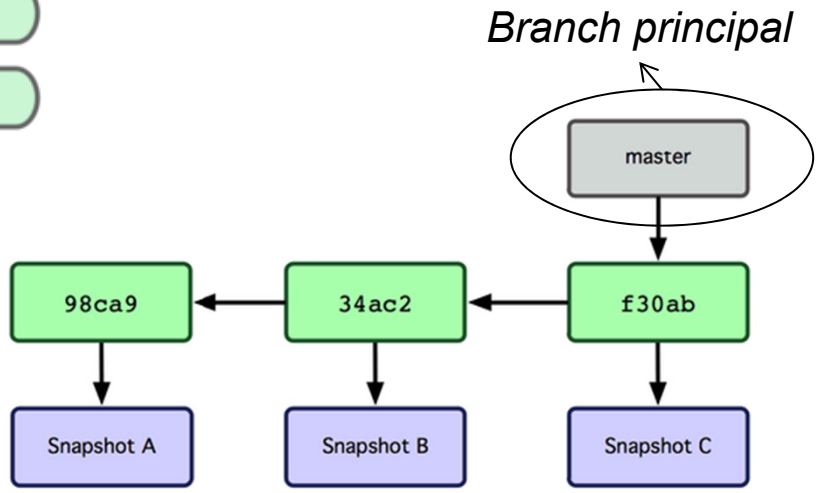


GITHUB

• Git modela sus datos como un conjunto de instantáneas de un mini sistema de archivos. Cada vez que se confirma un cambio *(commit)*, se hace una "foto" de los archivos en ese momento, y se guarda una referencia a esa instantánea:

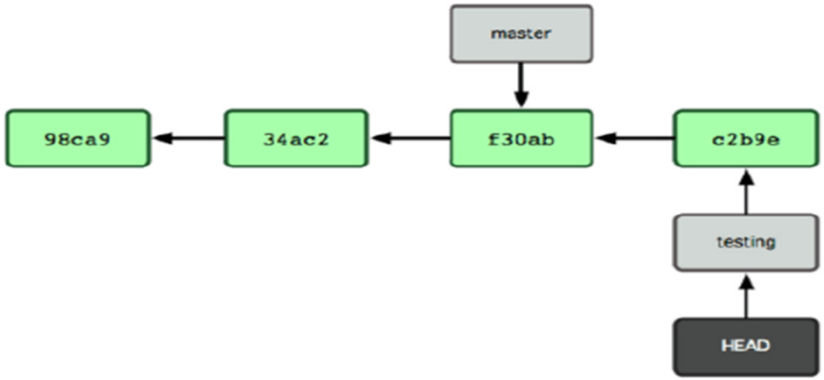


- Además, cada uno de los *commits* tiene un puntero a su *commit* anterior.

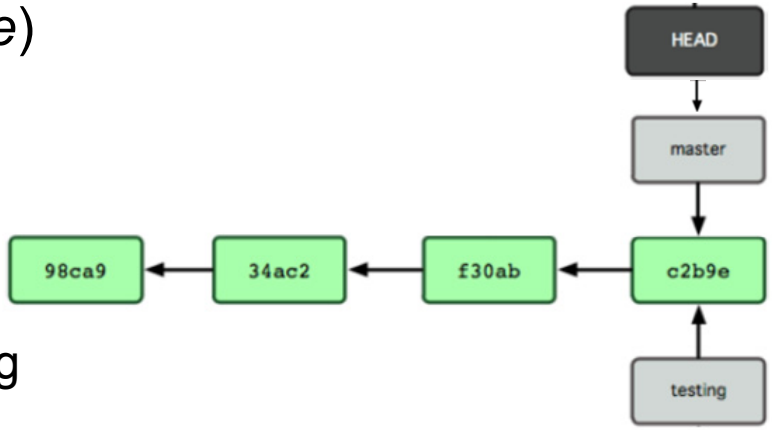


GITHUB

• *Un branch* o ramal es un puntero a un commit en concreto. Por otra parte, *HEAD* es un puntero al branch actual:



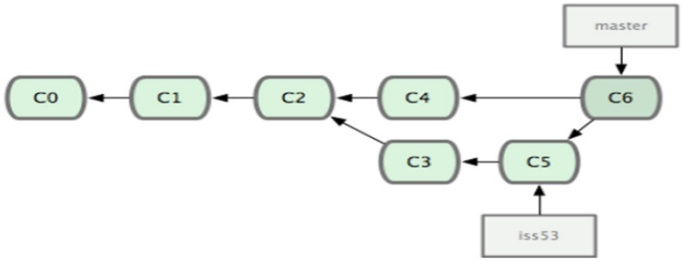
• Git permite trabajar con los ramales de forma que se pueden realizar cambios solo en uno de ellos y, una vez testeados los cambios se une el ramal al principal u a otro (operación *merge*)

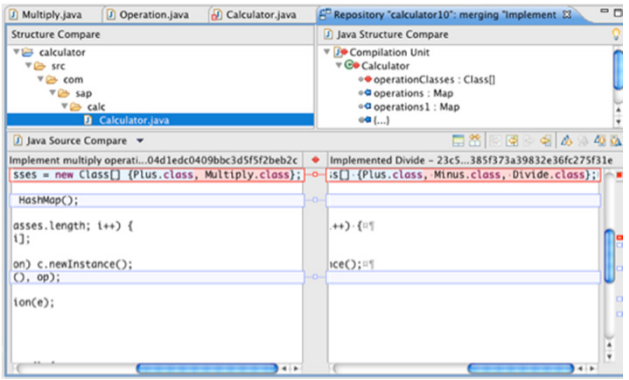


http://wiki.eclipse.org/EGit/User_Guide#Branching

GITHUB

• Puede ocurrir que la operación *merge* no se pueda hacer de forma automática debido a algún conflicto en las distintas versiones de los ficheros. En ese caso, podremos resolver el conflicto manualmente.





• La opción *rebase* también es útil en el caso de haber varias ramas con cambios:

```
      A topic
     /
D---E master
```

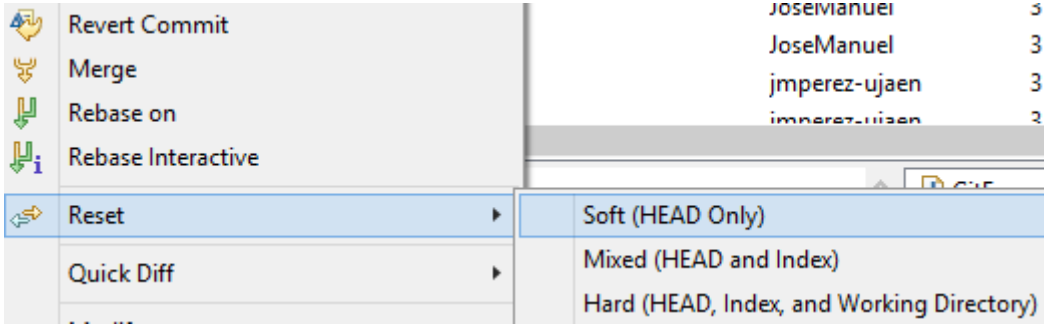
```
      A---B---C topic
     /
D---E---F---G master
```

```
                A'--B'--C' topic
               /
D---E---F---G master
```

http://wiki.eclipse.org/EGit/User_Guide#Branching

GITHUB

• Si queremos volver a un estado anterior, se nos ofrece la opción de hacer un *Revert Commit* o llevar el branch a un estado anterior mediante la opción *Reset.*



• Recomendación: antes de actualizar el repositorio remoto mediante un *push*, se recomienda hacer un *pull* previo, para resolver cualquier tipo de conflicto debido a los cambios que puedan ocurrir en el repositorio remoto.

http://wiki.eclipse.org/EGit/User_Guide#Branching

GITHUB

- Enlaces de utilidad:

- Guía de usuario de Egit:

  http://wiki.eclipse.org/EGit/User_Guide

- Proyecto Hello World para inicialización en plataforma GitHub:

  https://guides.github.com/activities/hello-world/