Sessió 3

ENTORN DE PROGRAMACIÓ AMB GIT + GITHUB

Programació PVV 2024-2025

GitHub

- GitHub és un servei d'allotjament de **repositoris** Git, el qual ofereix tota la funcionalitat de Git de control de revisió distribuït i administració de codi de la font (SCM) així com afegint les seves característiques pròpies.
- A diferència de Git, el qual és estrictament una eina de línia d'ordres, GitHub proporciona una interfície gràfica basada en web i escriptori així com integració del móvil.

Font: Viquipèdia

GitHub

- Repositori: “carpeta”/”contenidor” de tots els arxius d’un projecte de programació.
- Sign up amb el correu @institutperevives.cat
- Seguir el repositori de Programació

Git

- Git és un programari de sistema de control de versions dissenyat per Linus Torvalds, pensat en l'eficiència i confiabilitat de manteniment de versions d'aplicacions amb una enorme quantitat de fitxers de codi font.

Font: Viquipèdia

Git

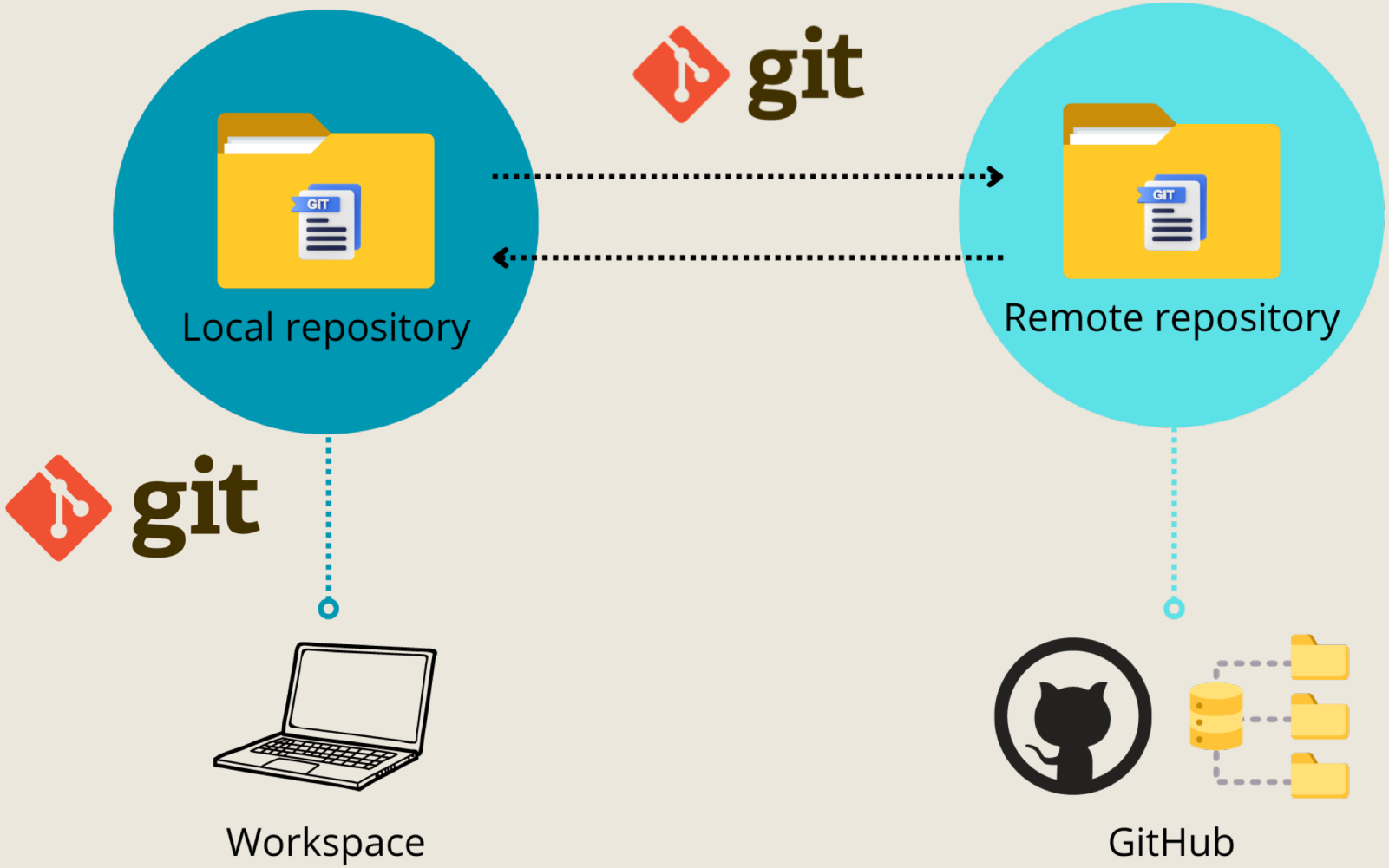
- Exemple en entorns professionals de programació:

Git

En el nostre cas:

- L'utilitzarem per tenir un control de versions i com a gestor d'arxius entre el repositori local (local repository) i el repositori de GitHub (remote repository).
- Cada projecte / pràctica tindrà la seva carpeta on a dins emmagatzarem els arxius corresponents.
- La carpeta que definim com a repositori local li inicialitzarem el git. Això ens permetrà tenir un control de totes les versions d'aquella carpeta i també gestionar la versió remota d'aquesta (repositori de GitHub).

Esquema de l'entorn



Conceptes bàsics de Git

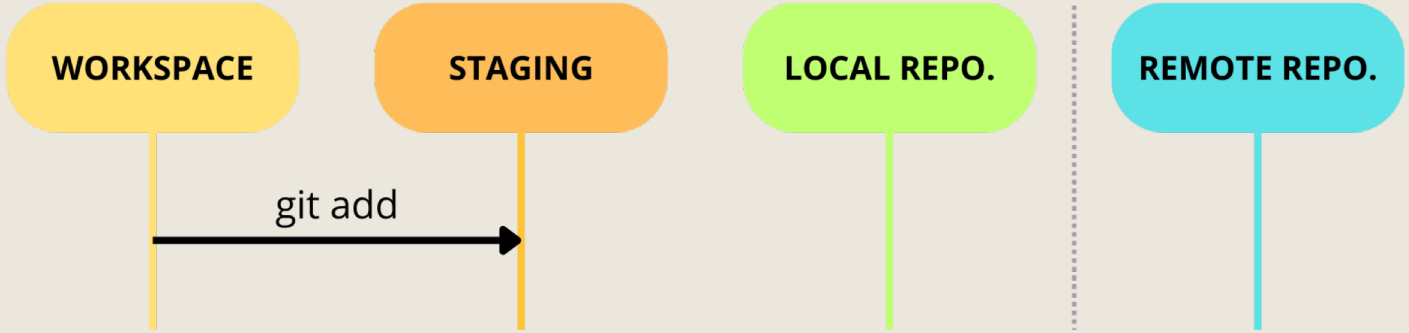
- Git es basa en snapshots (foto) del codi en un estat determinat, donat per un autor i una data.
- Un ***commit*** es un conjunt de canvis **guardats** en el repositori Git (en local) i té un identificador únic.
- Les branques (***branches***) permeten fer *commits* conservant en la branca principal l'últim commit que ens interessi. Com a mínim hi ha una branca principal predefinida anomanada *Master.*
- *Head* és el punter que ens indica l'últim *commit* de la branca on estem.
- *Remote* es refereix al repositori remot.

Workflow amb Git

Si treballem en local, inicialitzem la carpeta de treball (working directory).

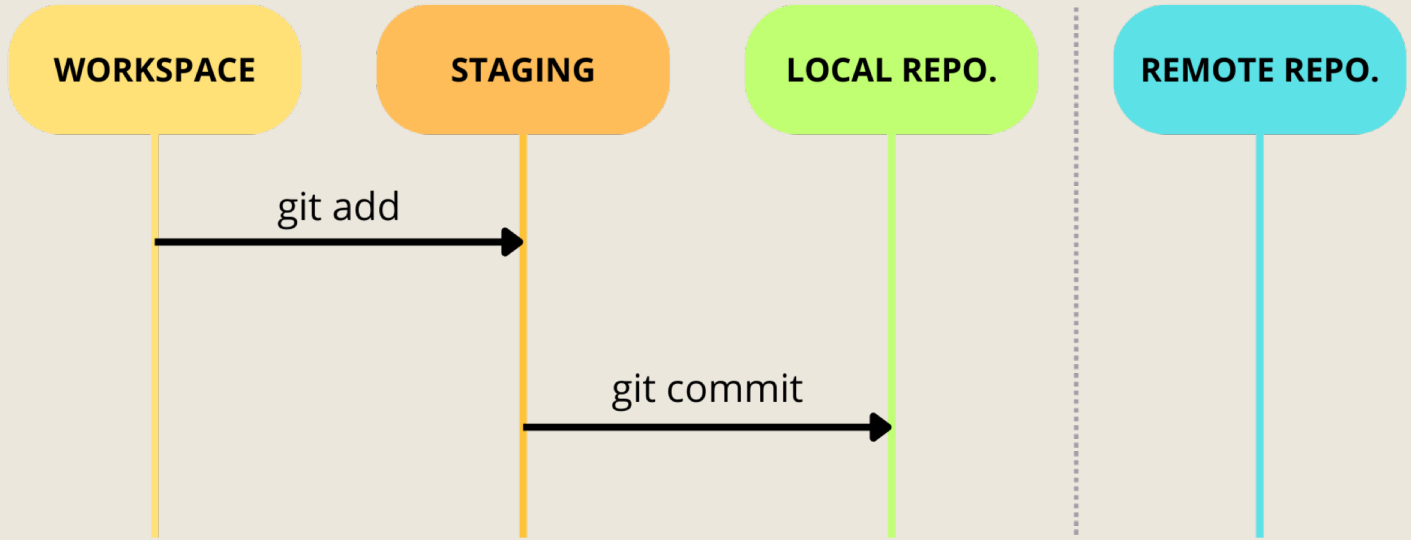
Podem treballar (editar fitxers) en la carpeta de treball.

Workflow amb Git



Amb la comanda *add* enviem els canvis a a *staging*, que és un estat intermig en el que es van guardant els canvis que volem enviar en el proper *commit*.

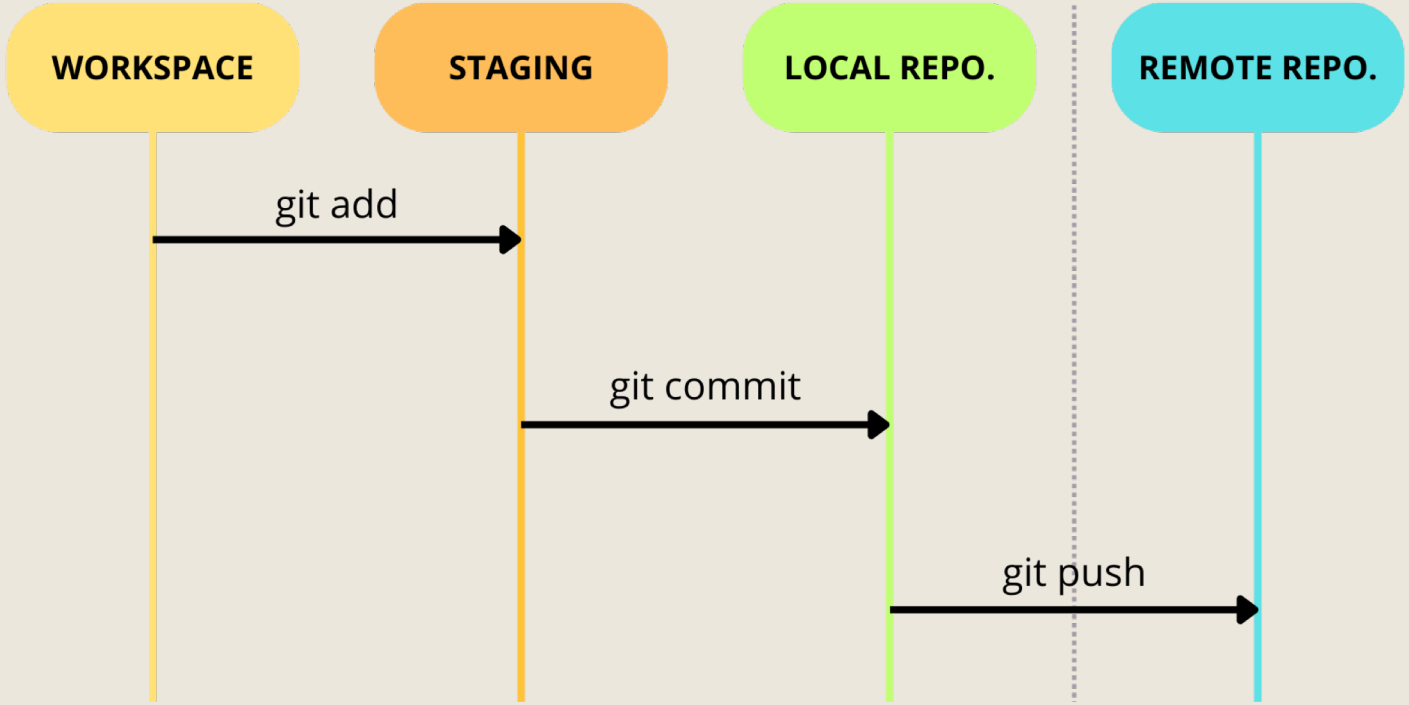
Workflow amb Git



Finalment amb el *commit* guardem els canvis que teniem guardats a *staging* al repositori local.

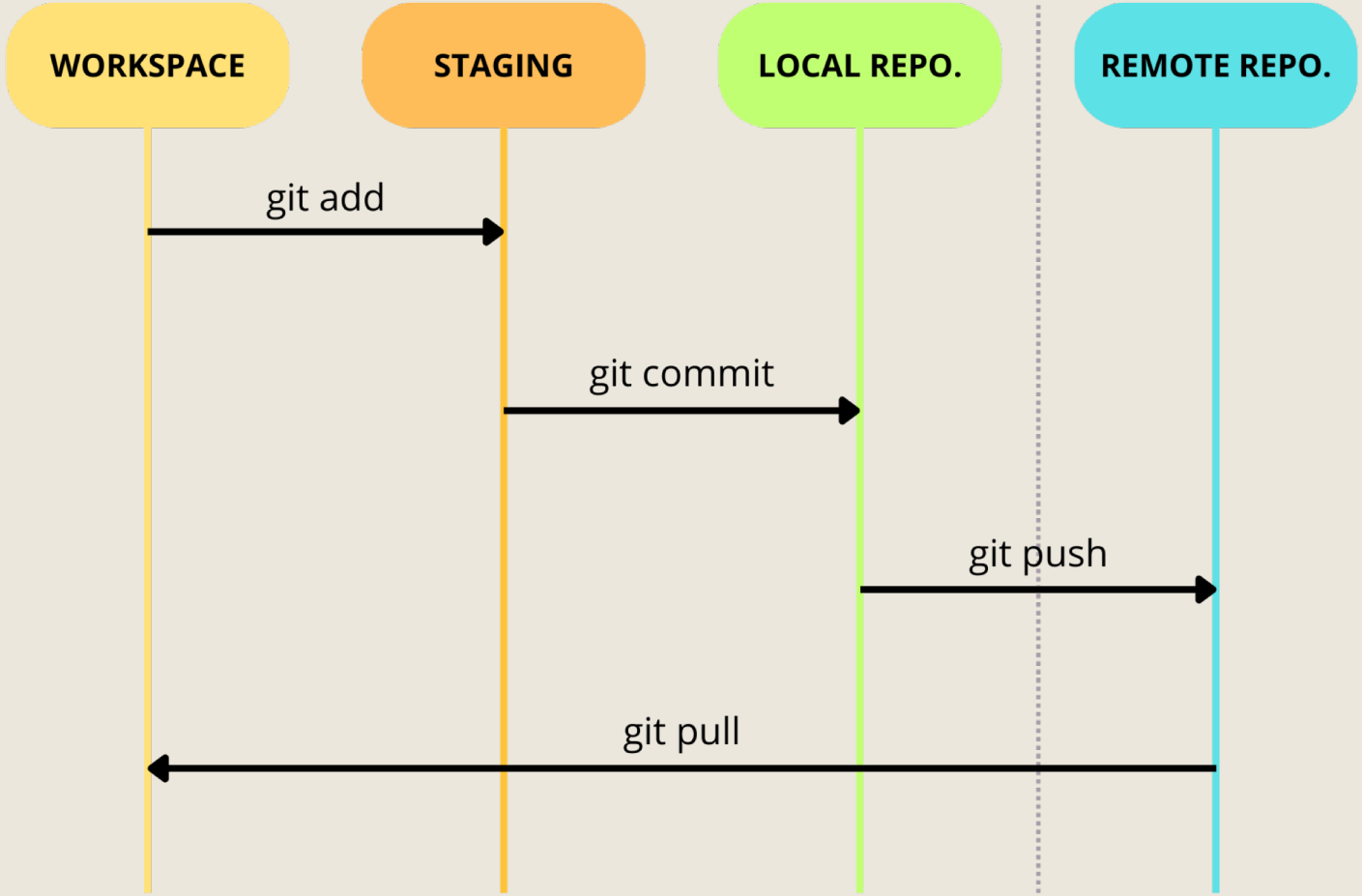
Workflow amb Git

Per guardar els canvis al repositori remot del GitHub utilitzarem la comanda *push.*



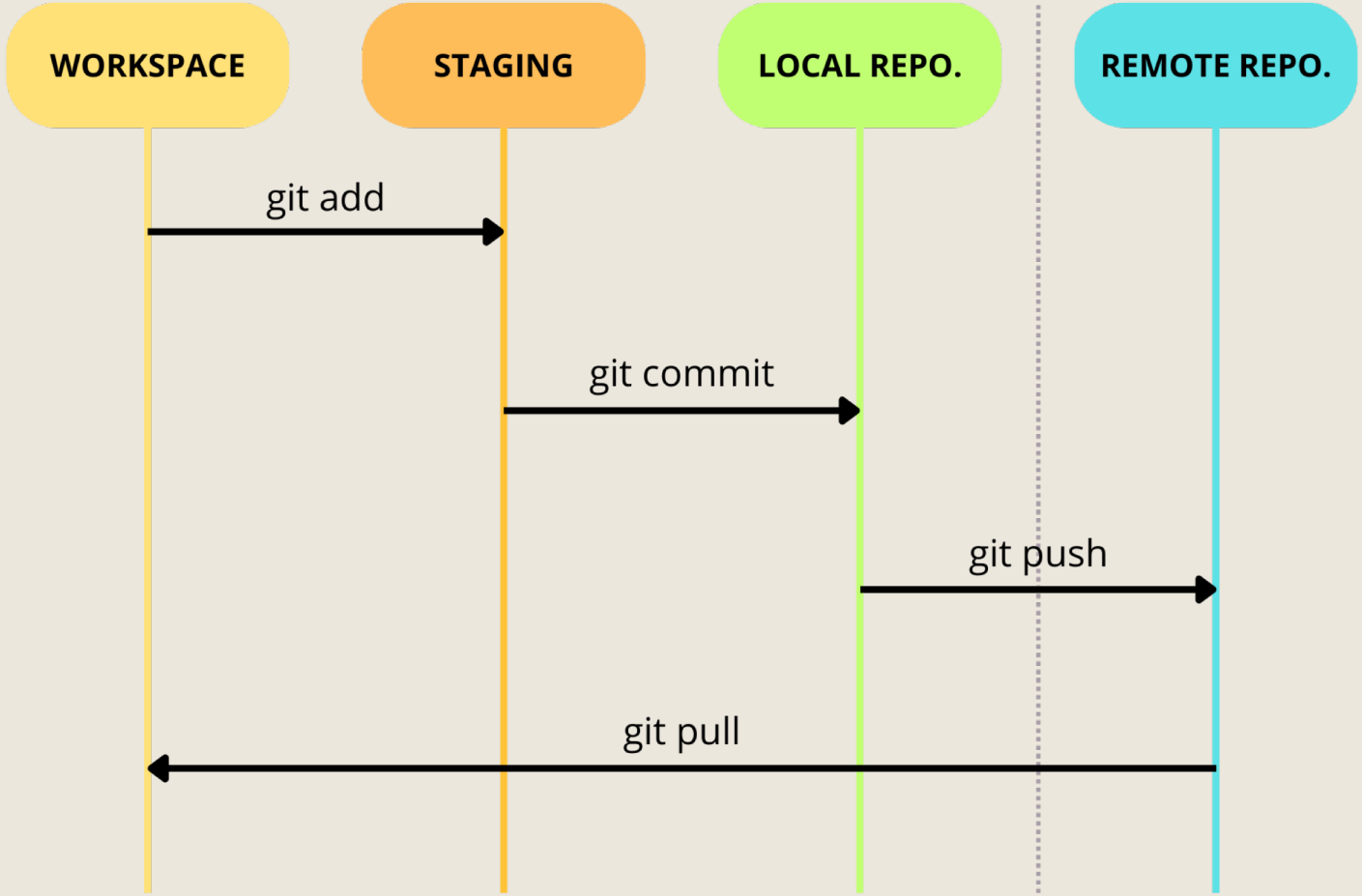
Workflow amb Git

Si treballem en diferents workspaces o no tenim el nostre workspace actualitzat amb el repositori remot, podem actualitzar els canvis amb la comanda *pull.*



Workflow amb Git

Per baixar-nos un nou repositori en el nostre Workspace, hem d'utilitzar la comanda *clone.*



Altres comandes amb Git

Aquestes són ordres habituals del Git usades en diverses situacions:

```
començar una àrea de treball (vegeu també: git help tutorial)
   clone     Clona un repositori a un directori nou
   init      Crea un repositori de Git buit o reinicialitza un existent


treballar en el canvi actual (vegeu també: git help everyday)
   add       Afegeix els continguts dels fitxers a l'índex
   mv        Mou o canvia de nom a un fitxer, directori o enllaç simbòlic
   restore   Restaura els fitxers de l'arbre de treball
   rm        Elimina fitxers de l'arbre de treball i de l'índex


examinar la història i l'estat (vegeu també: git help revisions)
   bisect    Troba per cerca binària el canvi que hagi introduït un defecte
   diff      Mostra els canvis entre comissions, la comissió i l'arbre de treball, etc
   grep      Imprimeix les línies coincidents amb un patró
   log       Mostra els registres de comissió
   show      Mostra diversos tipus d'objectes
   status    Mostra l'estat de l'arbre de treball
```

Altres comandes amb Git

```
fer créixer, marcar i ajustar la vostra història comuna
   branch    Llista, crea o suprimeix branques
   commit    Registra els canvis al repositori
   merge     Uneix dues o més històries de desenvolupament
   rebase    Torna a aplicar les comissions sobre un altre punt de basament
   reset     Restableix la HEAD actual a l'estat especificat
   switch    Commuta entre branques
   tag       Crea, llista, suprimeix o verifica un objecte d'etiqueta signat amb GPG


col·laborar (vegeu també: git help workflow)
   fetch     Baixa objectes i referències d'un altre repositori
   pull      Obtén i integra amb un altre repositori o una branca local
   push      Actualitza les referències remotes juntament amb els objectes associats


«git help -a» i «git help -g» llisten les subordres disponibles i
algunes guies de concepte. Vegeu «git help <ordre>» o
«git help <concepte>» per a llegir sobre una subordre o concepte
específic. Vegeu «git help git» per a una visió general del sistema.
```