



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

**1 DAM-SP
ENTORNS
CURS 20/21**

Control de versions amb GIT i GitHub

GIT és un sistema de control de versions distribuït.

GitHub és una plataforma online basada en GIT que permet que guardem els nostres repositoris GIT en un servidor remot i que varis usuaris puguin accedir i treballar de forma concurrent.

GitHub guarda una versió principal, suposadament estable, del projecte en el que estem treballant. En la màquina local tenim una còpia sobre la qual treballem, i quan tenim una nova versió actualitzem el repositori remot de GitHub amb els canvis que hem fet. El servidor guarda un historial de tots els canvis que hem anat fent.



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Control de versions amb GIT i GitHub

El primer pas és tindre un compte en GitHub. <https://github.com/>

Why GitHub? Team Enterprise Explore Marketplace Pricing

Search GitHub

Sign in Sign up

Built for developers

GitHub is a development platform inspired by the way you work. From **open source** to **business**, you can host and review code, manage projects, and build software alongside 50 million developers.

Username

Email

Password

Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more.](#)

Sign up for GitHub

By clicking "Sign up for GitHub", you agree to our [Terms of Service](#) and [Privacy Statement](#). We'll occasionally send you account related emails.



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB


**1 DAM-SP
ENTORNS
CURS 20/21**

Control de versions amb GIT i GitHub

Una vegada superem tots els tràmits (registre, verificació, etc.) ens apareix esta pantalla en la qual ja podem crear un repositori.


What do you want to do first?

Every developer needs to configure their environment, so let's get your GitHub experience optimized for you.




Start a new project
Start a new repository or bring over an existing repository to keep contributing to it.

Create a repository



Collaborate with your team
Improve the way your team works together and get access to more features with an organization.

Create an organization



Learn how to use GitHub
Get started with an "Introduction to GitHub" course in our Learning Lab.

Start Learning



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?

[Import a repository.](#)

Owner *



fidelDAM2021 ▾



Repository name *

prova



Great repository names are short and memorable. Need inspiration? How about **jubilant-disco**?

Description (optional)

Repositori de prova



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Control de versions amb GIT i GitHub

Ara podríem gestionar el nostre repositori manualment des de la línia de comandos tal com teniu als PDFs que vos he passat, però ací anem a veure com fer-ho des de **NetBeans**.

Des de NetBeans, anem a l'opció "Team" del menú principal, i després seleccionem "Clone".

Posem la URL del nostre repositori que hem creat en GitHub.

Seleccionem una carpeta local per a clonar el repositori remot.

Podem crear ja un projecte per a este repositori.



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Clone Repository

Steps

1. **Remote Repository**
2. Remote Branches
3. Destination Directory

Remote Repository

Specify Git Repository Location:

Repository URL: http[s]://host.xz[:port]/path/to/repo.git/

User: (leave blank for anonymous access)

Password: ☐ Save Password

[Proxy Configuration...](#)

Specify Destination Folder:

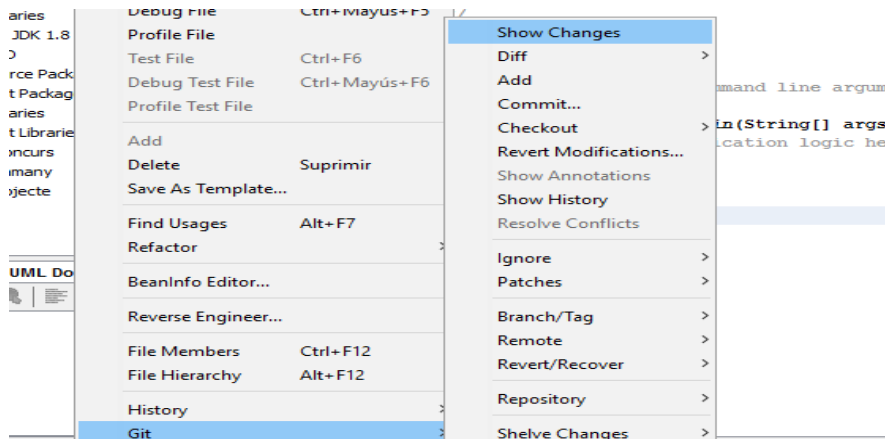
Clone into: [Browse...](#)



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Ara el nostre projecte tindrà, en el seu menú, un apartat anomenat Git on controlarem els canvis.





CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Ara fem un canvi en el projecte. En el main() posem un missatge que diga “Hola mundo”.

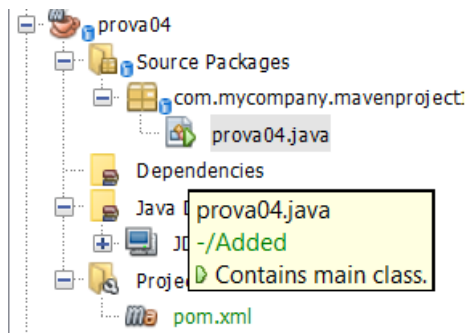
```
public class prova04 {  
    /**  
     * @param args the command line arguments  
     */  
    public static void main(String[] args) {  
        // TODO code application logic here  
        System.out.println("Hola, mundo");  
    }  
}
```




CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Fixeu-vos que el nom de l'arxiu apareix en verd en el menú. Això vol dir que no està actualitzat en el repositori.





CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Ara anem al nostre projecte, polsem el botó dret, en el menú seleccionem **“Git”** i dins del menú que s'obri, l'opció **“Commit”**.

Veureu que ens deixa seleccionar els fitxers que volem actualitzar, i també que dalt podem posar un comentari a la versió que anem a guardar. Podem posar, per exemple, *“Primera versió. Afegim el missatge 'Hola Mundo’”*. Polsem **“Commit”**.



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Commit Message:

Primera versió. Afegim el missatge "Hola, mundo"

Author: fidel <fidel@DESKTOP-U0GPNRV> Commiter: fidel <fidel@DESKTOP-U0GPNRV>

☐ Amend Last Commit

Files to Commit:

...	File	Status	Commit Action	Repository Path ▲
<input checked="" type="checkbox"/>	pom.xml	-/Added	Commit	mavenproject1\pom.xml
<input checked="" type="checkbox"/>	prova04.java	-/Added	Commit	...ycompany\mavenproject1\prova04.java



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Ara els nostres canvis s'han guardat en el repositori local. Encara no estan enviats al repositori remot.

Si tenim repositori remot, hem d'utilitzar l'opció **Remote => Push** del menú “**Git**” del nostre projecte.

Si hem configurat tot correctament, haurien d'eixir totes les dades per defecte i només hauríem de pulsar “Next”.



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Remote Repository

☒ Select Configured Git Repository Location:

origin:https://fidelDAM2021@github.com/fidelDAM2021/prova/

☐ Specify Git Repository Location:

Remote Name: ☒ Persist Remote

Repository URL:
http[s]://host.xz[:port]/path/to/repo.git/

User: (leave blank for anonymous access)

Password: ☐ Save Password

Proxy Configuration...



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

**1 DAM-SP
ENTORNS
CURS 20/21**

Ens demanarà que seleccionem un “Branch” (Rama) tant del projecte local com del repositori remot. Les rames serveixen per poder guardar diferents versions del mateix projecte. En un principi tenim una rama principal (sol anomenar-se `master`) on tenim la versió definitiva del projecte, i podem tindre una rama on guardem canvis que volem tindre el repositori però encara no en la versió definitiva.

Podem dir que les rames són per a guardar canvis provisionals, que encara no s'incorporen a la versió definitiva però que volem tindre en el repositori per seguretat o perquè treballem amb col·laboració d'altra gent.

Ara afegim un canvi al projecte i el guardem només en NetBeans.

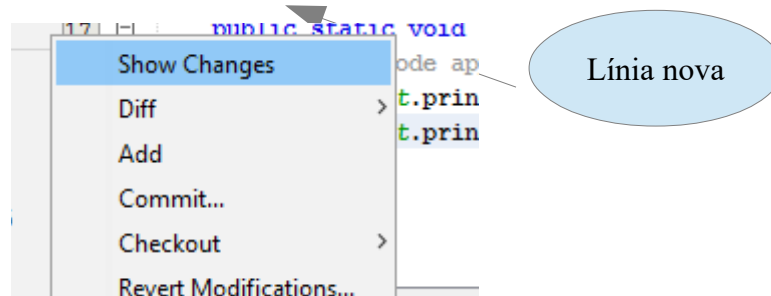
```
12  public class prova04 {  
13  
14      /**  
15       * @param args the command line arguments  
16       */  
17      public static void main(String[] args) {  
18          // TODO code application logic here  
19          System.out.println("Hola, mundo");  
20          System.out.println("=====");  
21      }  
22  }  
23
```

← ens marca la línia que hem afegit

← línia afegida

Mentre no fem commit al repositori local, podem veure els canvis que hi ha entre la versió guardada al repositori i la versió que tenim en NetBeans.

Hem d'anar al projecte, buscar “Git” i “Show changes”.

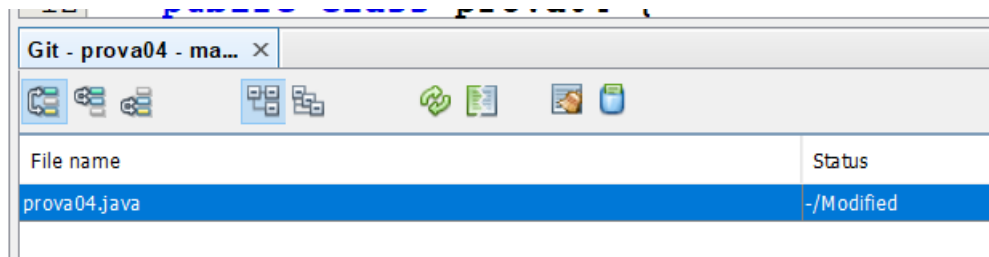




CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Ens apareixerà una pantalla així:





CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

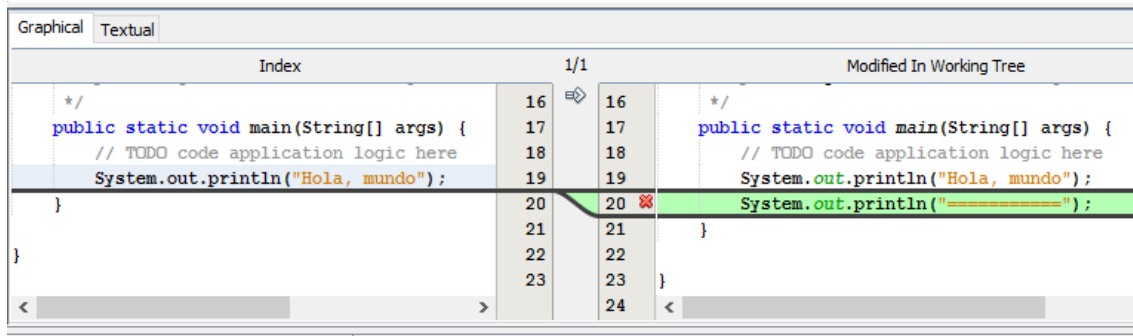
Ens diu que el fitxer **prova04.java** està modificat i no actualitzat en Git.

Si polsem el botó  ens donarà més detalls.



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21



Fixeu-vos com Git ens diu quin canvi hem fet en el projecte i encara no està actualitzat en el repositori de versions.

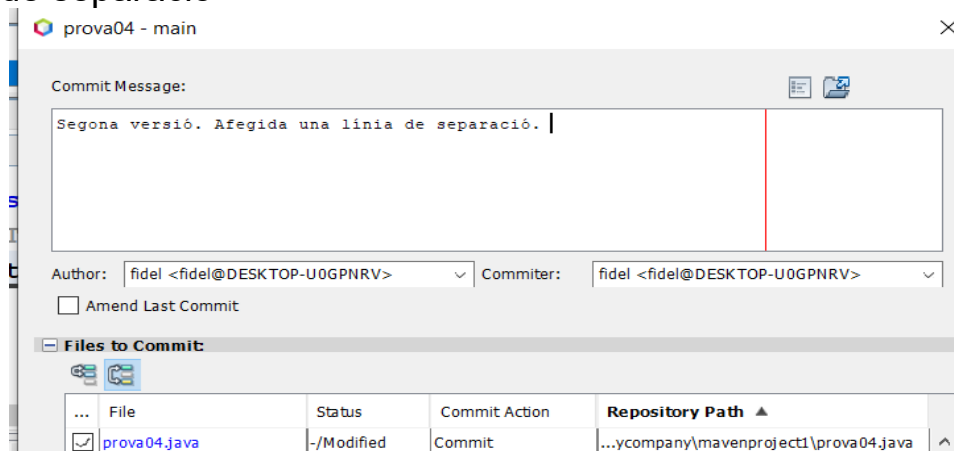
Anem a actualitzar-lo fent “commit”.



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Posem un comentari, per exemple “Segona versió. Afegida una línia de separació”

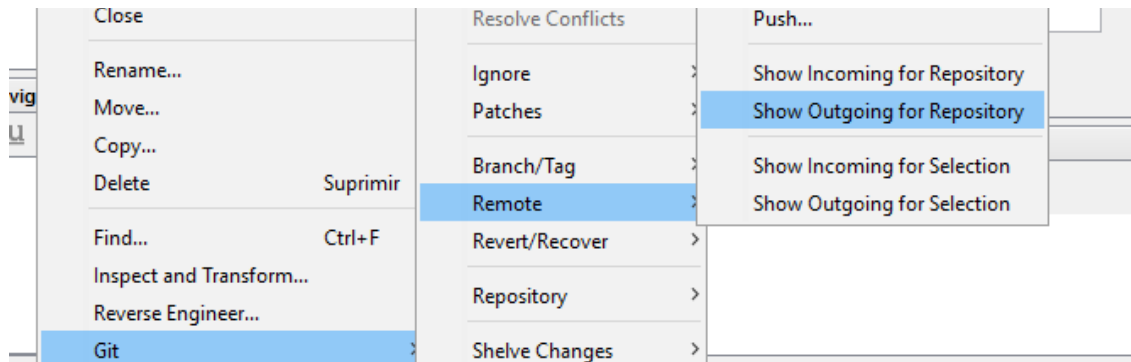




CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

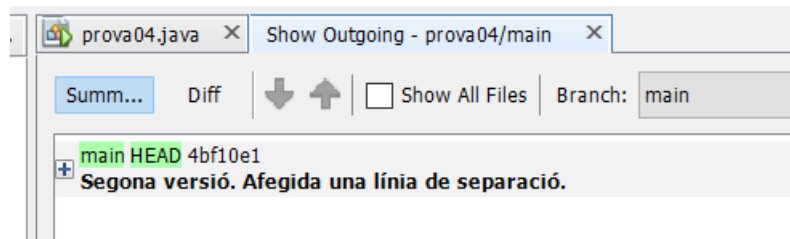
Mentre no fem Remote => Push, els canvis estan en el repositori local, per a control nostre, però no estan en el repositori remot. Per tant, altres usuaris no tindrian accés a eixos canvis. Podem veure quins canvis estan guardats en local i no en remot si anem al projecte, obrim el menú amb el botó dret, seleccionem



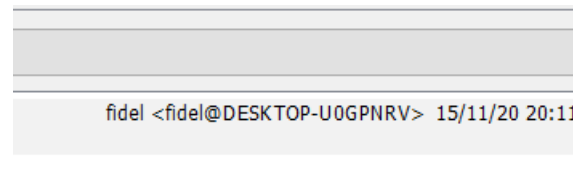


CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21



També ens diu quin usuari ha fet el canvi, per si hi ha més gent treballant en el repositori local.



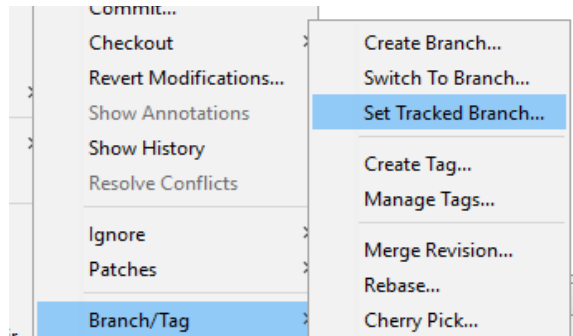


CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Les rames (Branch) funcionen de la següent manera:

- 1) Podem crear una nova rama
- 2) Podem treballar en eixa rama, de manera que podem pujar els canvis al repositori però no afecten a la rama principal
- 3) Podem fer “merge” i actualitzar els canvis d'una rama sobre la rama principal






CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Per exemple, anem a fer un canvi provisional que no volem guardar en la versió definitiva. Primer actualitzem el repositori remot en tots els canvis anteriors. Després fem el següent canvi:



```
public static void main(String[] args) {  
    // TODO code application logic here  
    System.out.println("Hola, mundo");  
    System.out.println("=====");  
    // estem estudiant passar el missatge a valencià  
    System.out.println("Hola, món");  
    System.out.println("=====");  
}
```




CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

I creem una rama nova en el repositori, anomenada “provisional”

Create Branch

Branch Name:

Revision:

Commit ID:

Author:

Date:

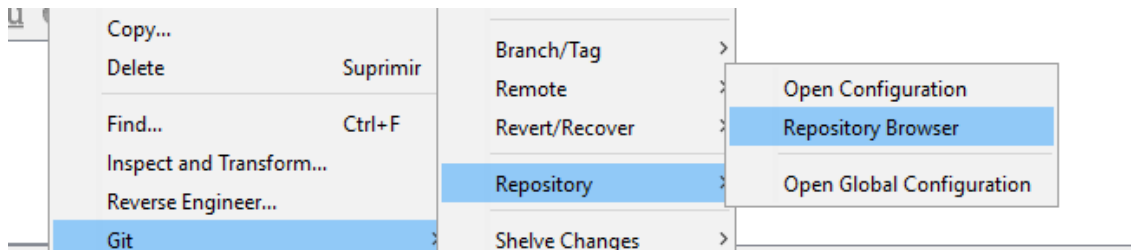
Message:



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Si anem a

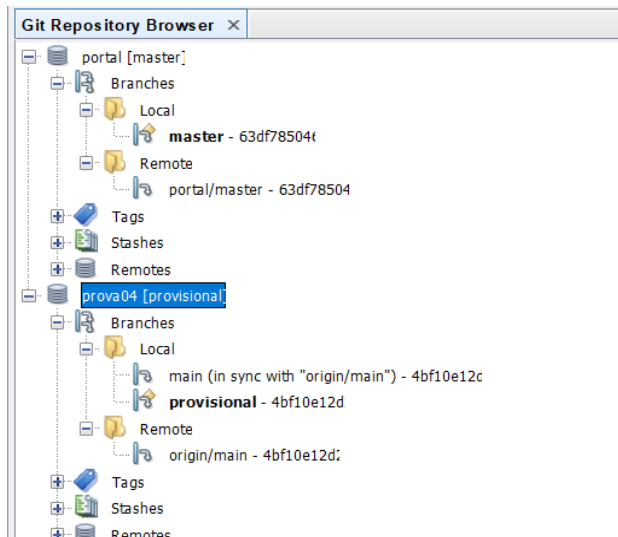


Podem veure la nostra estructura de repositoris



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21



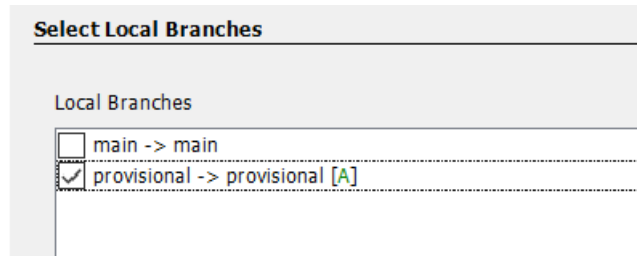


CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Ara:

- Quan fem “commit” en local, podem triar la rama “master” o la “provisional”

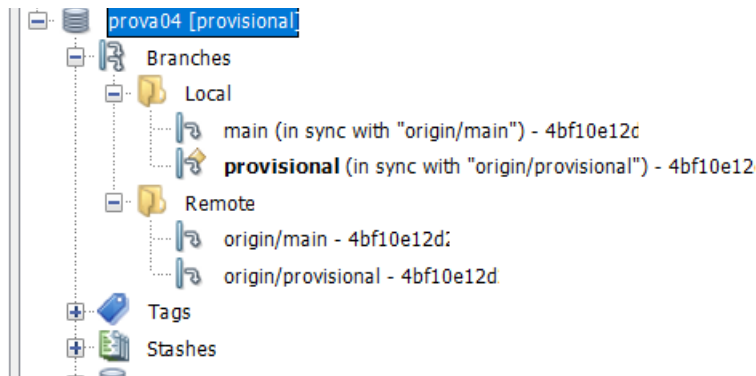




CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

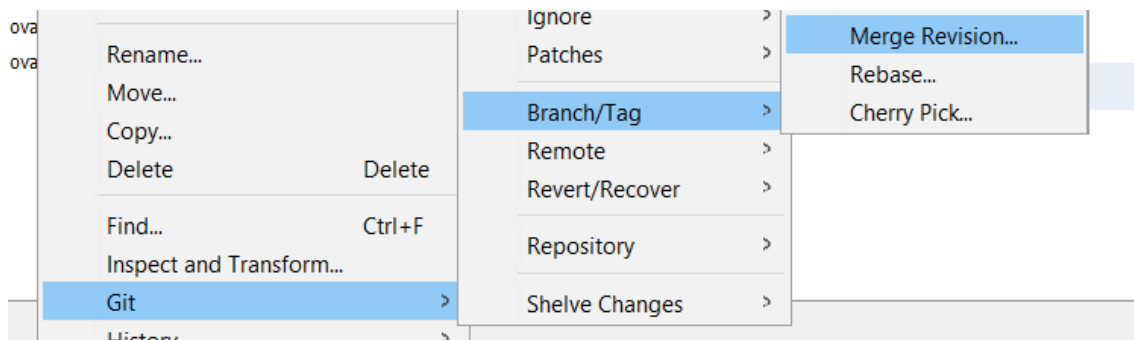
Si pugem la branca “Provisional” i després mirem com se queda el repositori, veurem:



CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

- Quan tenim clar els canvis que hi ha en “provisional”, podem fer “merge” i passar-los a la rama principal

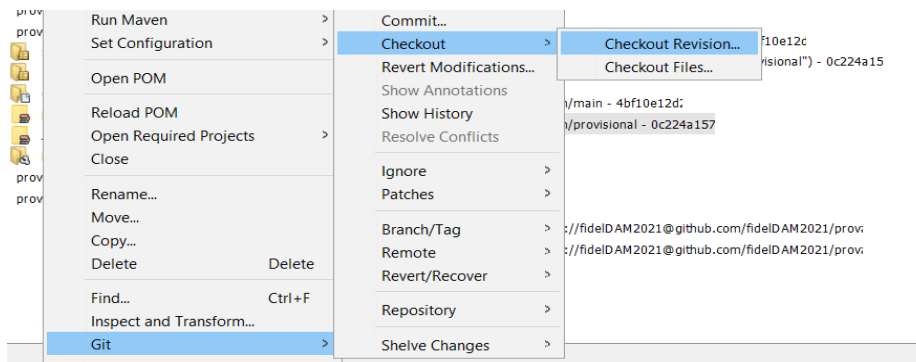




CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Podem utilitzar l'opció “checkout” per seleccionar una branca determinada i actualitzar els arxius locals al seu contingut

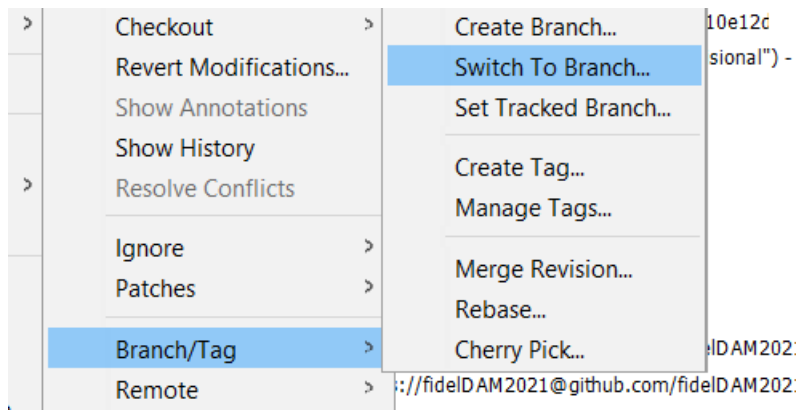




CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

Amb l'opció “Switch to branch” passem a treballar amb la branca seleccionada





CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

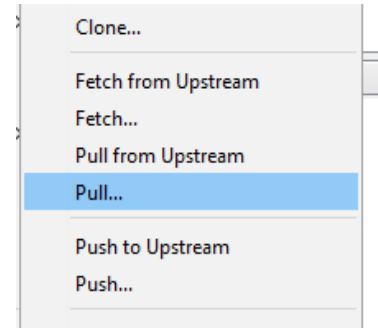
1 DAM-SP
ENTORNS
CURS 20/21

Actualitzar el nostre repositori local des del repositori remot

Quan hi ha més gent treballant en un projecte, haurem d'actualitzar el nostre repositori local amb els canvis que la gent ha anat pujant al repositori remot.

Funciona tot igual, però l'opció que hem de triar és "Pull"

**Projecte => Git =>
Remote => Pull**





CONTROL DE VERSIONS AMB NETBEANS, GIT I GITHUB

1 DAM-SP
ENTORNS
CURS 20/21

També podem utilitzar l'opció **Fetch**, en lloc de **Pull**, per comprovar quins canvis hi ha al repositori remot que nosaltres en local encara no tenim, però sense modificar el nostre repositori local.

Més informació (en anglès) en

<https://netbeans.org/kb/docs/ide/git.html>