LUNDS UNIVERSITET

Versionshantering med Git och GitHub

EDAA60 DATORER OCH DATORANVÄNDNING

Versionshantering? Vad är det?

- Hjälper oss hantera komplexiteten det innebär att hålla reda på alla filer som hör till ett projekt – källkod, dokumentation, etc.
- Låter oss gå tillbaka till äldre versioner av filerna om nödvändigt.
- Gör det möjligt för oss att arbeta parallellt med filerna i projektet utan att komma i vägen för varandra.

Versionshanteringssystem

- Programvara som lagrar alla versioner som har existerat under projektets gång i någon form av databas.
- Centraliserade system: Subversion, CVS, etc. En central databas, ett *repositorium* (*repo*), håller reda på alla filer. Utvecklare *checkar ut* en lokal kopia av filerna och arbetar med dem. När arbetet är klart checkar utvecklaren in filerna igen (kopierar dem till det centrala repot) varvid en ny version av projektet skapas.
- Distribuerade system: Git. Finns inget centralt repo. I stället *klonar* man ett repo (gör en komplett kopia av repot). Ändringar kan hämtas (pull), eller överföras till (push), andra repon.

Git och GitHub

- Git är en programvara med vars hjälp man kan skapa nya repon, klona ett repo från en annan dator (via nätet), arbeta med det på sin lokala dator samt hämta/överföra ändringar.
- GitHub är en webbtjänst (bland flera andra liknande) som kan användas för att lagra git-repon samt utföra git-operationer på dem. Gratis att använda.
- Beroende på ett projekts komplexitet så kan arbetsprocessen se olika ut. Laborationen visar hur Git och GitHub kan användas i ett mindre projekt (t.ex. för en inlämningsuppgift) med ett fåtal deltagare.
- I kommande kurser, t.ex. EDAF45 Programvaruutveckling i grupp , kommer ni att stöta på mer avancerad användning av Git.

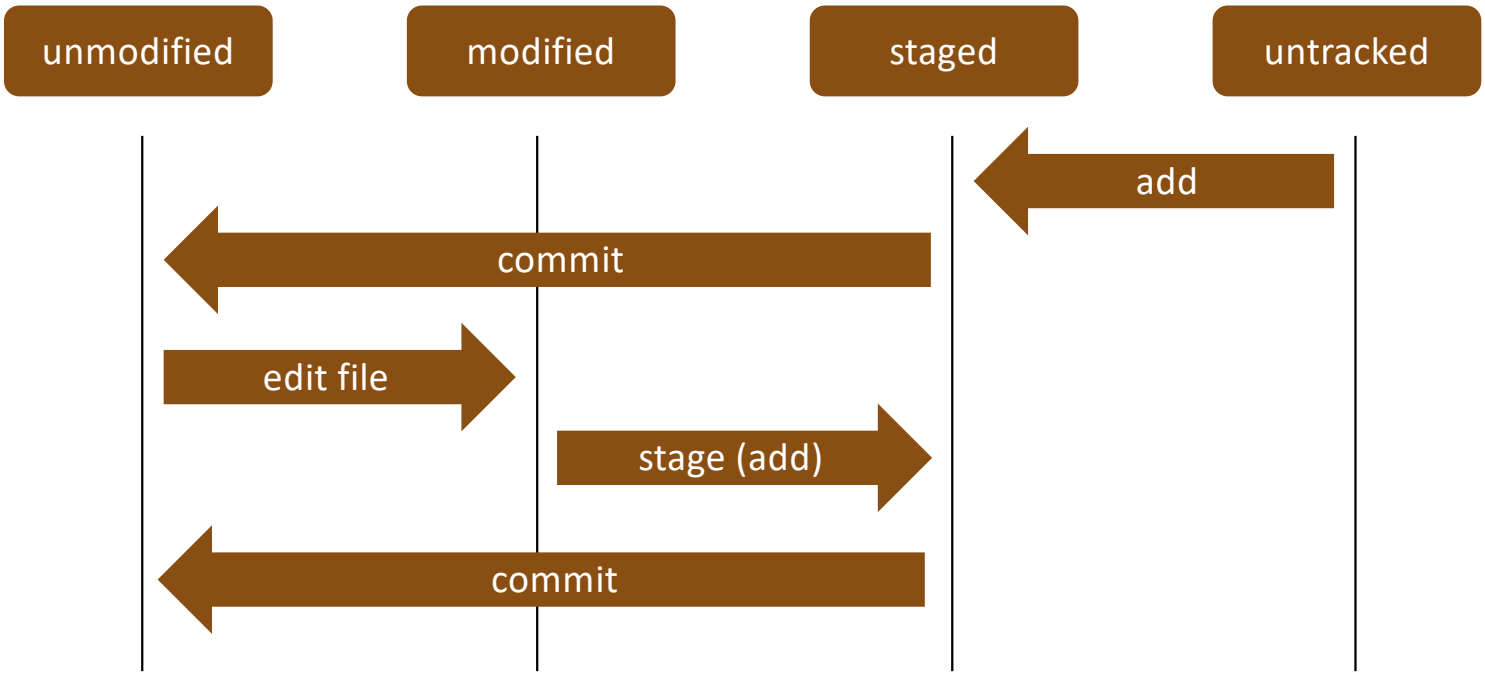
Om Git

- Skapad av Linus Torvalds, Linux skapare, 2005, för att hantera utvecklingen av Linuxkärnan.
- Mål:
  - Snabbhet
  - Stödja massivt parallell utveckling (tusentals samtidiga aktiviteter)
  - Helt distribuerat
  - Kunna hantera stora projekt effektivt

Arbeta med ett lokalt repo

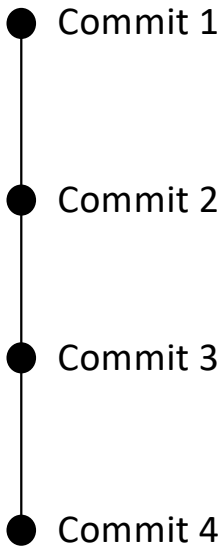
1. Skapa ett repo på din egen dator (eller klona ett existerande)
2. Lägg till eller modifiera filer. När du har något som fungerar kan du skapa en ny version av projektet. Filerna är nu *modifierade*.
3. Välj ut vilka (oftast alla) av de modifierade filerna som ska ingå i den nya versionen av projektet. De är nu *utvalda* (eng. *staged*).
4. Skapa den nya versionen genom att göra en s.k. *commit* (svensk term?). Nu lagras en ny uppdaterad version av alla filerna i repots databas.
5. Upprepa från punkt 2.

Filers livscykel



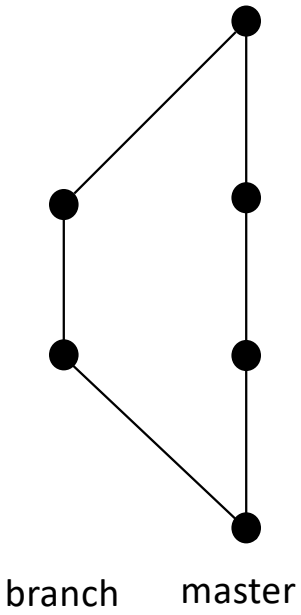
Commit

Varje commit skapar en ny ögonblicksbild (snapshot) av hela projektet som vi kan återvända till om nödvändigt.



Grenar (branches)

Man kan skapa parallella utvecklingslinjer som sedan sammanfogas igen (merge).



Merge

- När två grenar slås ihop kan konflikter uppstå om ändringarna i de två grenarna är motstridiga.
- Ofta klarar Git av att själv göra en merge, men ibland kan manuellt ingripande vara nödvändigt.

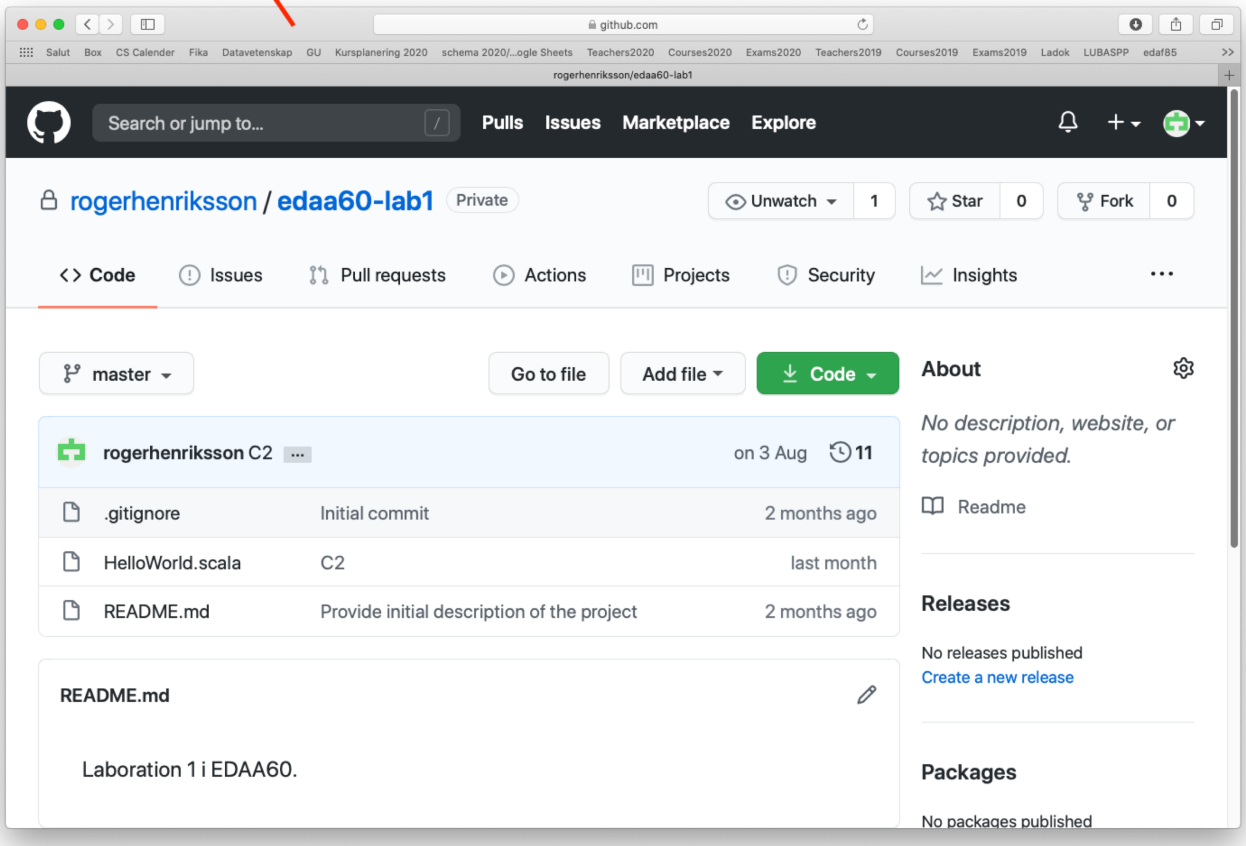
```
<<<<<<< HEAD:index.html
<div id="footer">contact : email.support@github.com</div>
=======
<div id="footer"> please contact us at support@github.com </div>
>>>>>>> iss53:index.html
```

- Efter en merge, testa att allt fungerar som det ska och gör commit.

Klonade repo

- Ett repo som ligger på GitHub eller på en dator nåbar via nätet kan kopieras, klonas, till din lokala dator. Du får då en komplett kopia av repot.
- Om repot du klonade senare uppdateras och du vill hämta ner de ändringar som gjorts gör du en s.k. *pull*.
- Om du har skrivrättigheter i originalrepot kan du ladda upp dina lokala ändringar genom att göra en s.k. *push*.
- Har du *inte* skrivrättigheter, men ändå vill ladda upp dina ändringar måste du göra en s.k. *pull request*. Ägaren till repot blir då informerad om dina ändringar och kan välja att hämta dem med en pull.

GitHub


github.com
rogerhenriksson/edaa60-lab1
Search or jump to...
Pulls
Issues
Marketplace
Explore
rogerhenriksson / edaa60-lab1
Private
Unwatch 1
Star 0
Fork 0
Code
Issues
Pull requests
Actions
Projects
Security
Insights
master
Go to file
Add file
Code
rogerhenriksson C2
on 3 Aug
11
.gitignore
Initial commit
2 months ago
HelloWorld.scala
C2
last month
README.md
Provide initial description of the project
2 months ago
README.md
Laboration 1 i EDAA60.
About
No description, website, or topics provided.
Readme
Releases
No releases published
Create a new release
Packages
No packages published

GitHub

- Gratis
- Lagring av personliga repon
- Kan utföra git-operationer via webbgränssnitt: Skapa, klona, uppdatera repon.

- Publika repon – lämpliga för open-source-programvara och sådant ni vill visa för andra, t.ex. blivande arbetsgivare.
- Privata repon – lämpliga för laborationer/inlämningsuppgifter och liknande. Undvik att andra stjäl era lösningar och lämnar in dem som sina!

Enkel arbetsprocess för litet projekt

1. Skapa ett repo på GitHub som gemensam lagringsplats som alla projektdeltagare har skrivrättigheter till.
2. Varje deltagare klonar repot till sin egen lokala dator.
3. Deltagarna arbetar med det lokala repot tills det innehåller något som de vill dela med sig av till övriga deltagare. Då gör de en *push* som överför ändringarna till repot på GitHub.
4. De andra deltagarna kan när de sedan vill ta del av ändringarna genom att göra en *pull*.

Merge vid push/pull

- Om någon annan redan gjort en push när vi försöker ladda upp våra ändringar måste vi först integrera våra ändringar med de ändringar som gjorts i det centrala repot. Git upptäcker detta åt oss.
- Det gör vi genom att göra en pull som hämtar ner den senaste versionen av projektet och sammanfogar det med våra ändringar (merge). Därefter kan vi göra push på nytt.

Vad händer nu?

1. Läs avsnittet i scalakompendiet (Björn Regnell) om Git.
2. Se avsnitt 1.1, 1.2, 1.3, 1.6 samt 1.7 i youtubeserien ”Git and GitHub for Poets”.
3. Läs kapitel 1, 2 och 3.1-3.2 i Pro Git. (Chacon/Straub).

Länkar till materialet finns på kurshemsidan under ”Föreläsningar”.

På laborationen får ni arbeta er igenom ett scenario som visar hur ni kan använda Git och GitHub för ett litet projekt.

LUNDS
UNIVERSITET