Git - Branches

- Branch
- Merge
- Remote branch

Git – Branches (Arbetsflöde)

- Arbeta i (den inbyggda) Master branchen med t ex en webbapplikation.
- Vi vill arbeta med en ny User Story eller annan mindre del. Vi skapar då en ny branch som vi arbetar med för den delen.
- Vi arbetar på den nya delen som har en egen branch. Gör förändringar och commits som vanligt.
- Därefter gör vi en merge för att slå ihop förändringarna med Master branchen.
- Sträva efter att ha kortlivade branches som raderas när den är färdig och vi gjort en merge med Master branchen.

Git – Branch och checkout

- **Skapa en ny branch och gör checkout för att börja arbeta på den nya branchen**

$ git branch htmledit

$ git checkout htmledit

- **Man kan också göra branch och checkout i samma kommando**

$ git checkout -b htmledit

Git – Branch och Checkout

- **Gör förändringar när du arbetar på den nya branchen och gör commits som vanligt**

$ git commit –a –m ”Lagt till Grid layout på indexsidan”

- **När du är klar gör du en merge då vi slår ihop förändringarna med master branchen.**
  (Se alltid till att du har aktuell kopia från Github genom att köra git pull när du är på master branchen innan du ska göra en merge)
  Först måste vi växla till Master branchen (main) med checkout.
  Därefter gör vi en merge med den branchen vi har arbetat med.

$ git checkout main
$ git merge htmledit

Git – Branch och checkout

- **En branch man är klar med vill man inte ha den kvar. Man strävar alltid efter att ha kortlivade brancher och det som är aktuellt på Master branchen (main).**
  Nu raderar vi branchen htmledit med branch -d

$ git branch -d htmledit

Git – Branch och Merge

Git Branching - Basic Branching and Merging

https://git-scm.com/book/en/v2/Git-Branching-Basic-Branching-and-Merging

https://www.atlassian.com/git/tutorials/using-branches

Git - Branches

- För att visa vilken branch man är på skriver man git branch

  $ git branch

Git – Branch Managment

Git Branching – Branch Management

https://git-scm.com/book/en/v2/Git-Branching-Branch-Management

Git – Merge conflicts

- När man har arbetat med en ny branch och ska slå ihop den med Master branchen kan det uppstå Merge confict s. Det beror på att någon i Master branchen har arbetat i samma fil(-er) samtidigt så att filen ser olika ut på samma rader. Då får vi en konflikt och vi måste bestämma vad vi ska behålla (hur filen ska se ut).

Git – Merge conflicts

Gå till Basic Merge Conflicts i denna länken

https://git-scm.com/book/en/v2/Git-Branching-Basic-Branching-and-Merging

https://www.atlassian.com/git/tutorials/using-branches/merge-conflicts

Git – Remote Branches

- Om vi vill samarbeta med en ny branch måste vi göra den branchen tillgänglig på Github. Det gör vi med git push och skriver vilken branch vi vill pusha.

  $ git push origin htmledit

Git – Remote Branches

- Därefter måste den som vill arbeta på den nya branchen hämta hem den.
  Först måste vi hämta referensen till serverversionen genom fetch.

  $ git fetch origin

- Därefter måste vi hämta ut själva filerna för branchen så att våra lokala filer är uppdaterade.
  $ git checkout -b htmledit origin/htmledit

Git – Remote Branches

- När du har kopplat samman den lokala med branchen på Github (i föregående steg) använder du push som vanligt för att skicka över förändringar (commits) till den branchen du arbetar på.

  $ git push

Git – Remote Branches

- Vi kan också radera en Remote branch (på Github)

  $ git push origin --delete htmledit

Git – Remote Branches

Git Branching – Remote Branches

https://git-scm.com/book/en/v2/Git-Branching-Remote-Branches

Övning 1 (Lokal Branch)

Arbeta i grupper. Utgå från övningen med ett enkelt ”Webbprojekt” med html och CSS. Alternativt gör ett Github repo med liknande upplägg för att repetera alla steg som vi gjorde i den övningen.
Först ska vi träna på att skapa en ny branch som (en person) arbetar med lokalt därefter göra en merge och slår ihop den nya branchen med Master branchen. Den nya branchen kan t ex innehålla en flexbox, tabell på indexsidan eller någon annan html-sida.

- Skapa en ny branch
- Gör ändringar i den nya branchen
- När du är klar med ändringarna merga ändringarna från den nya branchen i master
- Radera den nya branchen när du är klar

Övning 2 (Merge conflict)

Arbeta i grupper. Utgå från övningen med ett enkelt”webbprojekt” med html och CSS. Alternativt gör ett Github repo med liknande upplägg för att repetera alla steg.

- Nu ska vi skapa en Merge konflikt och det gör vi genom att någon arbetar i samma fil t ex indexsidan samtidigt som någon arbetar i en ny branch.
- Innan personen som arbetar med den nya branchen och gör en merge med Master branchen måste man se till att ha en aktuell kopia från Github genom git pull när man är i Master branchen.
- Därefter gör man en merge som tidigare. Skillanden är nu att vi kommer att få en Merge konflikt och då behöver man se till att filen är så som vi vill ha den och därefter göra en merge. (Jag visar alla steg på genomgången)

Övning 3 (Remote Branch)

Om tid finns testa också att arbeta med Remote Branch då ni arbetar tillsammans på en ny branch. Steg man behöver göra finns på tidigare slides (som jag också gick genom på lektionen)

Men börja med att arbeta med egna lokala Branches först.

Inför projektet

- Skapa ett Vue vite project som en i gruppen lägger upp på Github. Se till att .gitignore innehåller det ni behöver ha med (filer som inte ska delas med andra).
- De andra klonar ner projektet från Github.
- Gå till mappen där Vue vite-projektet finns som ni har klonat ner. Där behöver ni köra
  npm install
  för att Vue vite-projektet ska gå att köra på sin dator.

Inför projektet

Tips: När ni startar upp projektet börja först med att arbeta på den inbyggda Main-branchen därefter arbetssättet med att skapa en ny branch när behov finns. Ni känner i gruppen vilket som passer er bäst.