

INSTALL GIT+GITHUB

self guiding guide

BRUG SAMME COMPUTER OG BROWSER TIL ALT!

Hen mod slutningen af denne guide, i integrationen af GitHub og IntelliJ, får du en masse mails med links som du skal følge.

Det er vigtigt at du åbner disse links på din computer hvor du kører IntelliJ, i den browser hvor du er logget ind på Github!

Så lad være med at bruge din mobil til at tjekke mail, og lad være med at bruge mere end én browser,

DOWNLOAD GIT

DOWNLOAD AF GIT

Der er stor forskel på hvordan git downloades og installeres på Mac og Windows, så hop til den del af guiden der matcher din computer.

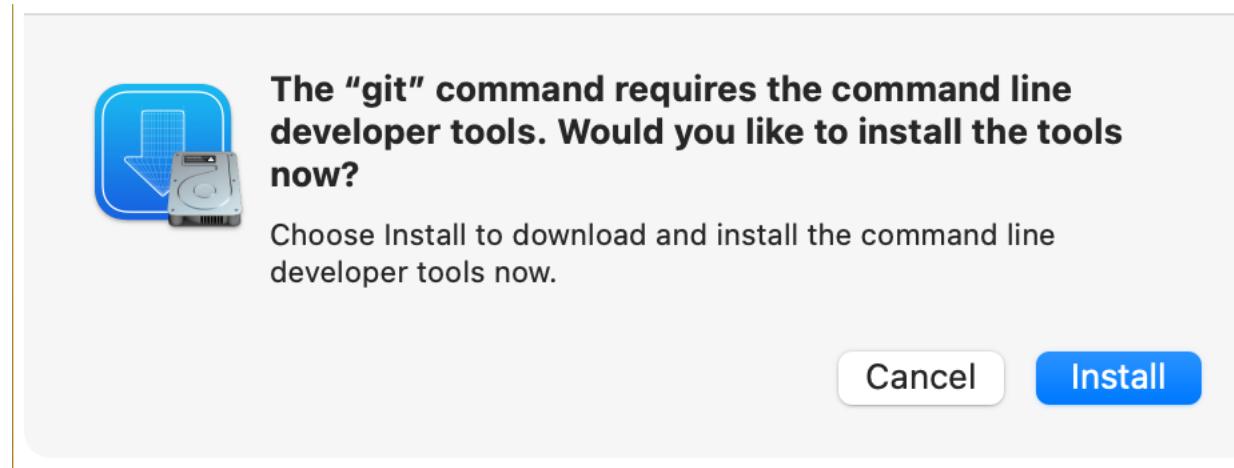
DOWNLOAD GIT (MAC)

DOWNLOAD GIT (MAC)

Den nemmeste made at downloade Git på er:

1. Start Terminal applikationen på din Mac
2. I terminal vinduet skriv git –version
3. Hvis ikke Git er allerede installeret, vises et vindue hvor man vælge at installere command line tools, som inkluderer Git.
4. Vælg Install

% git --version



TEST INSTALLATIONEN AF GIT

For at bekræfte at git er installeret korrekt – med den korrekte version – skal du åbne en ny terminal. (du kan ikke bruge den hvor du lige installerede det!)

Skriv `git --version` og tryk enter (altså *git*, mellemrum, to bindestreger og ordet *version*)

Det skulle gerne resultere i et svar noget lignende dette:

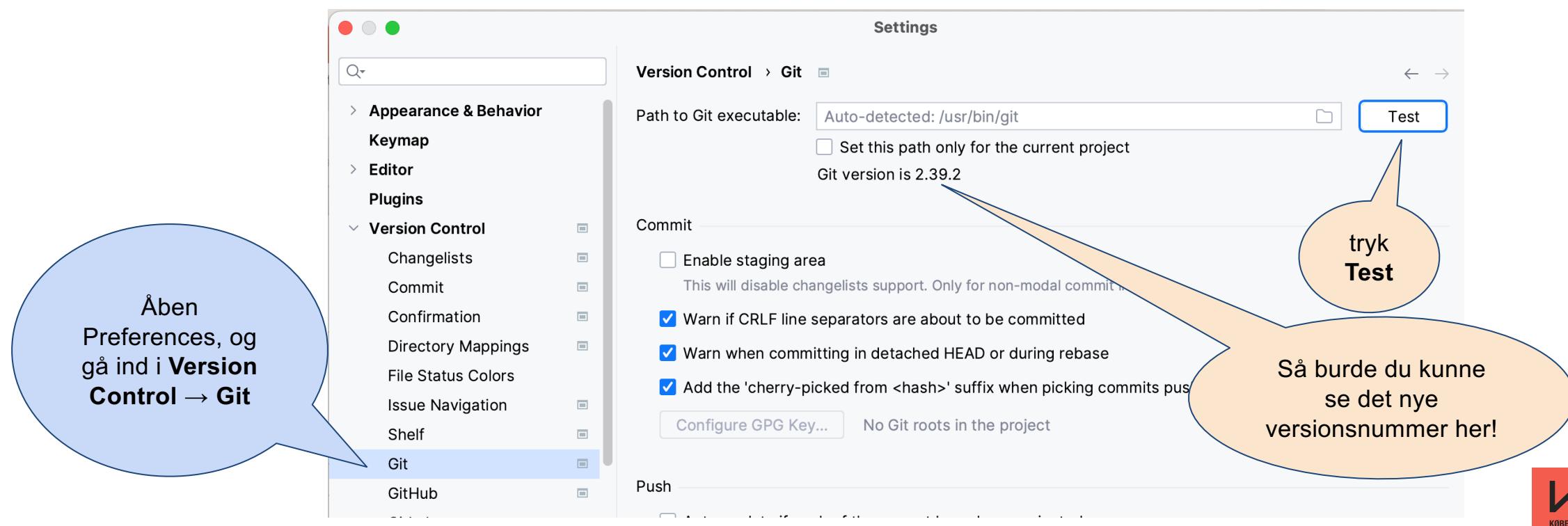
```
git version 2.39.2 (Apple Git-143)
```

(Dette er Apples fork af Git og er typisk et par versioner bagud)

OVERDØV DEN GAMLE VERSION AF GIT I INTELLIJ

Mac har ofte installeret Git som standard, som regel i en ældre udgave, og nogle gange vælger IntelliJ at bruge den forkerte. Det skal måske rettes.

Så åbn IntelliJ, og tjek at den bruger den version du lige har installeret



De næste mange sider handler om installation på Windows,
så spring over dem, og gå videre til [GitHub](#)

DOWNLOAD GIT (WINDOWS)

DOWNLOAD GIT (WINDOWS)

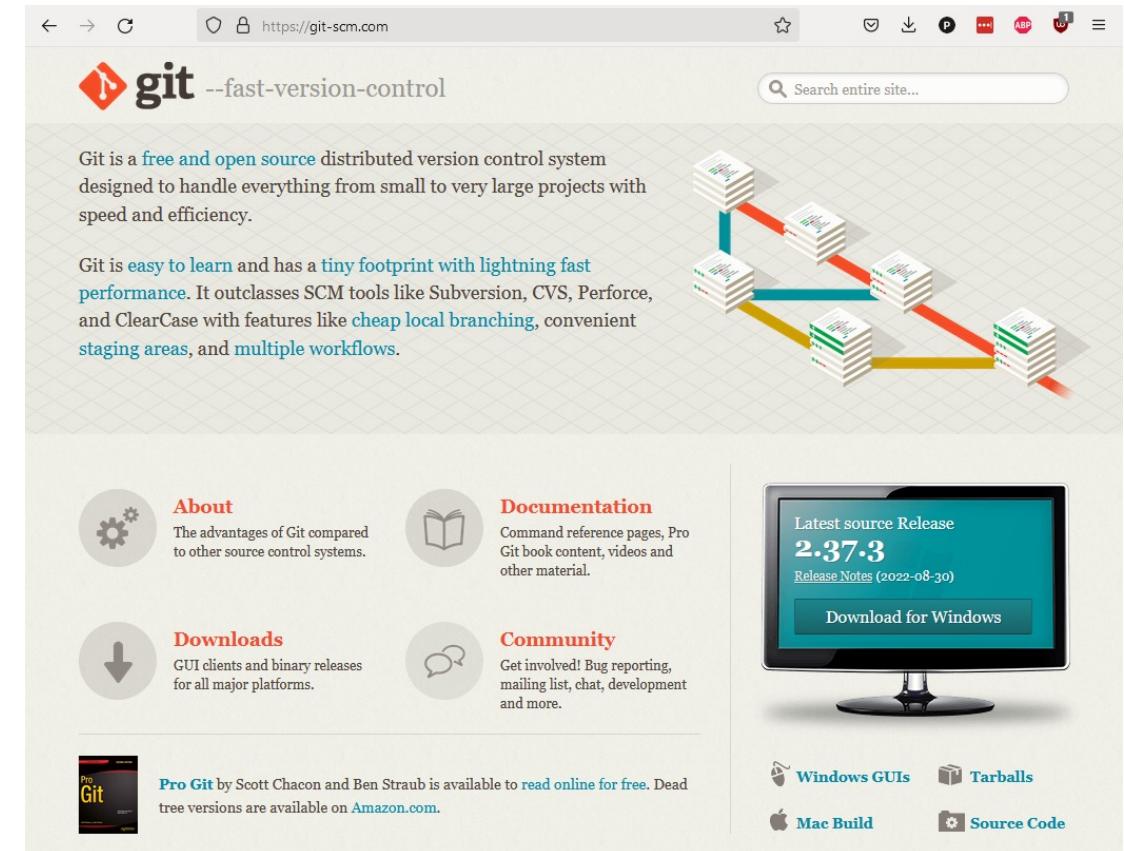
Download af git på Windows, er ret ligetil.

Gå ind på <https://git-scm.com/> og tryk på Download for Windows knappen.

Hvis downloaden ikke starter automatisk, så tryk på det allerførste "Click here to download" link.

Det giver dig et installationsprogram som du kan starte som normalt.

Følg resten af guiden her for at finde ud af hvad du skal vælge i de mange dialogbokse.



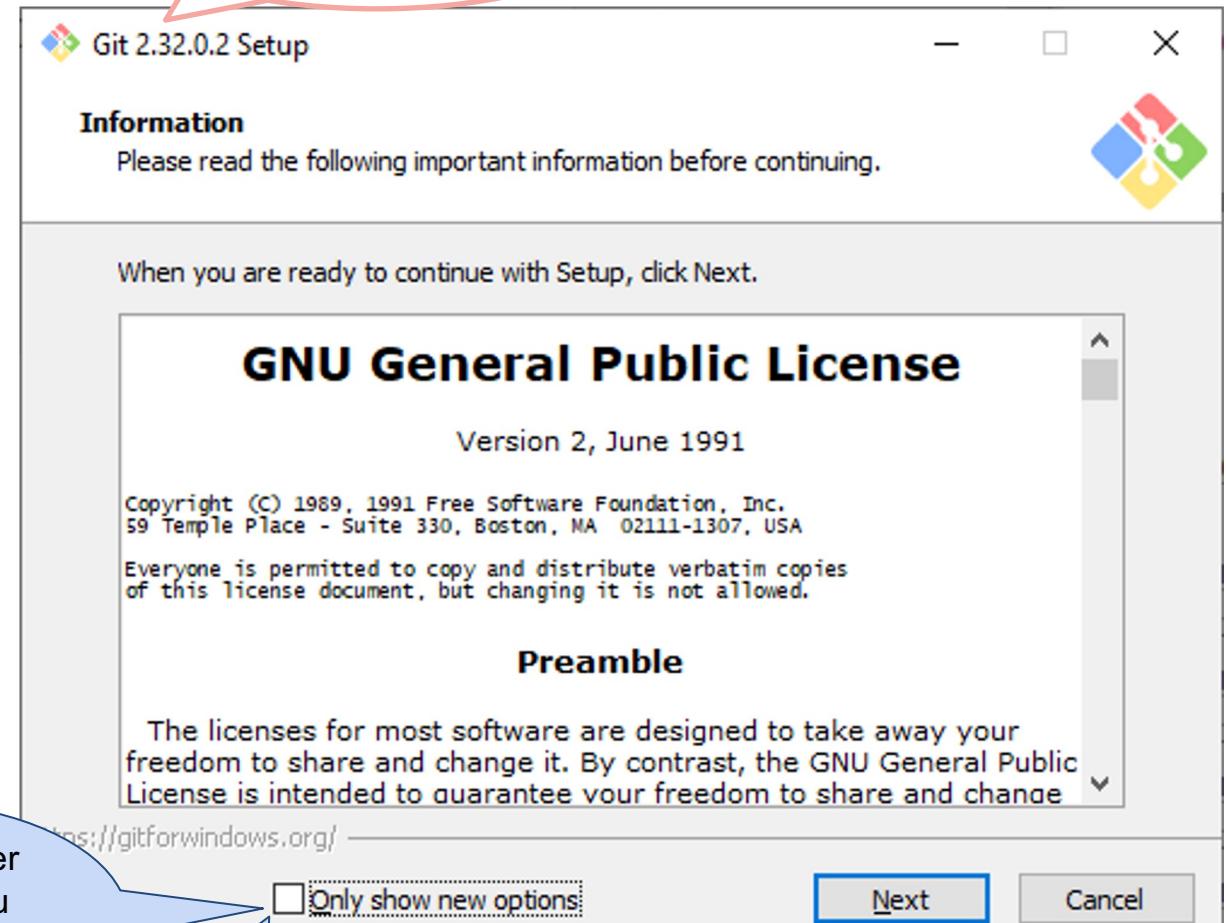
INSTALLER GIT (WINDOWS)

INSTALLER GIT (WINDOWS)

Git på Windows spørger om vildt mange detaljer når det skal installeres.

For det meste kan man blot klikke Next, men der er nogle enkelte sider hvor det kan være en fordel at vælge noget andet - derfor denne guide!

NB: Nogle billeder er fra en ældre version
– tag dig ikke af det



Denne boks ser du kun hvis du opdaterer fra en ældre version.

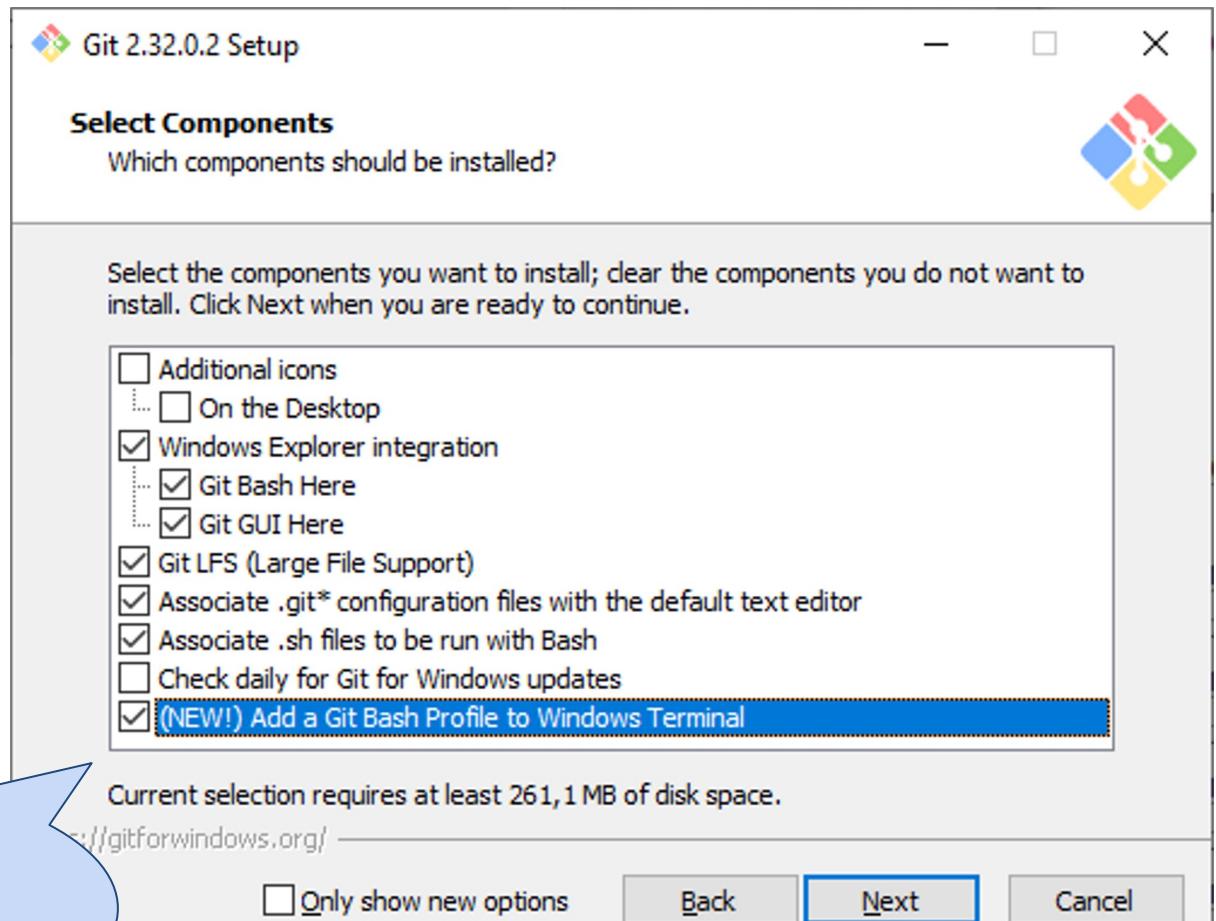
Slå den fra for at se alle sider!

INSTALLER GIT (WINDOWS)

(2/15)

Select Components

Du kan lade dem stå som standard, men slå alt med "Git Bash" **til** - og slå Check for updates **fra**.



Egentlig kun relevant
hvis du bruger
Windows Terminal -
men slå den til alligevel

INSTALLER GIT (WINDOWS)

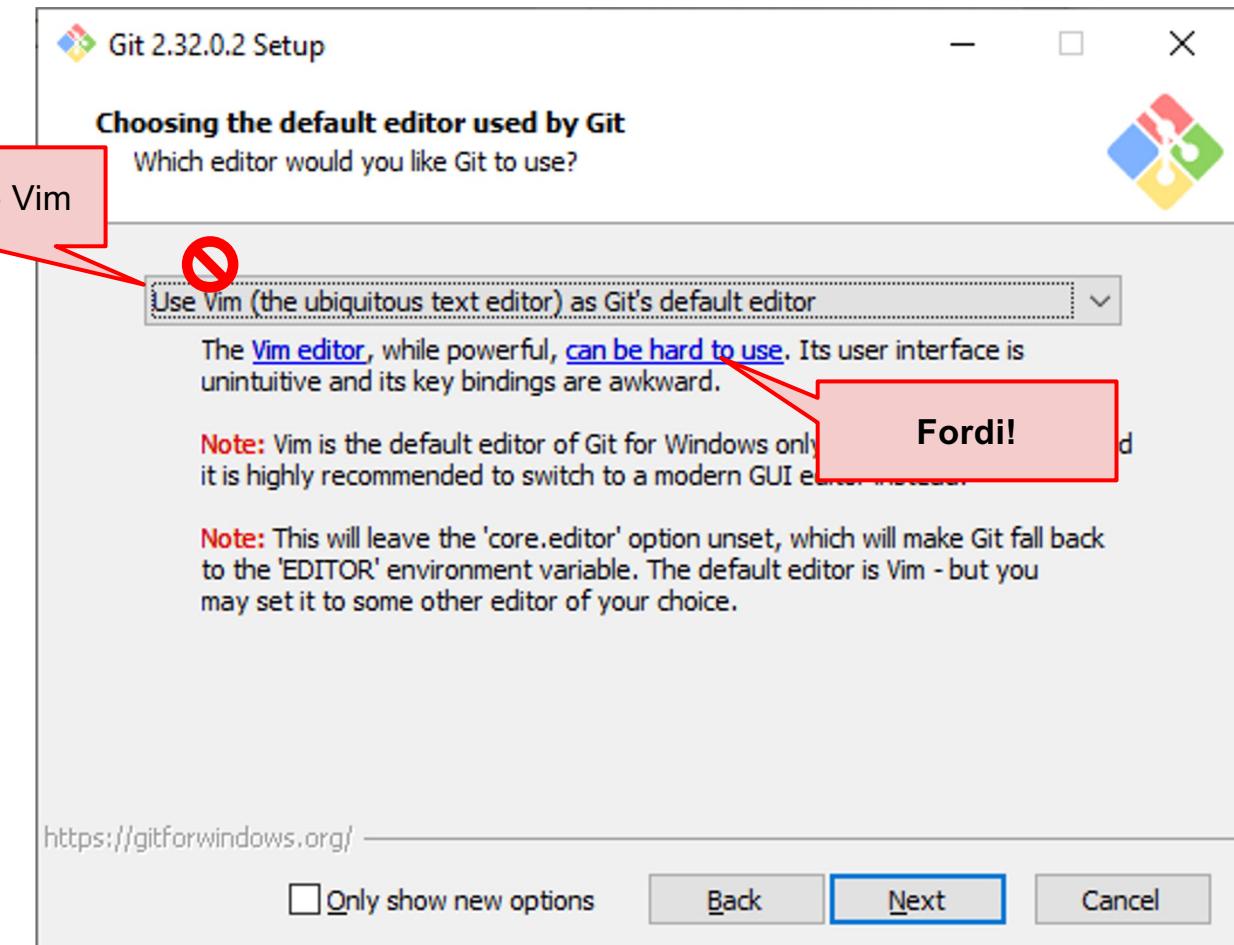
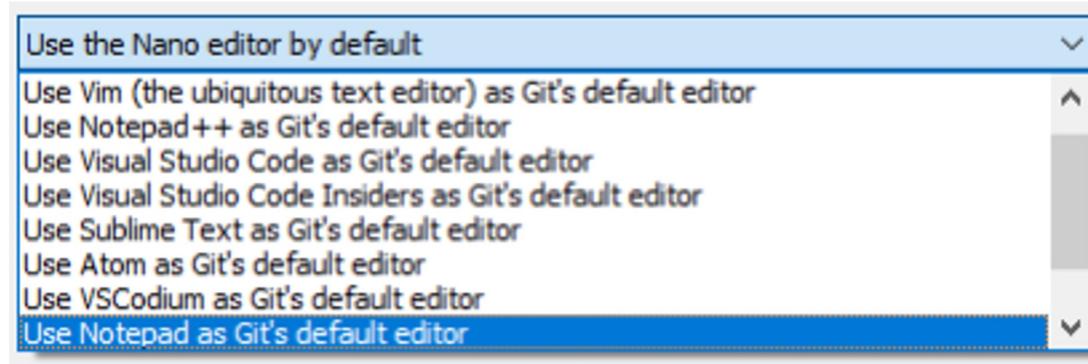
(3/15)

Editor

Vælg noget andet end Vim!

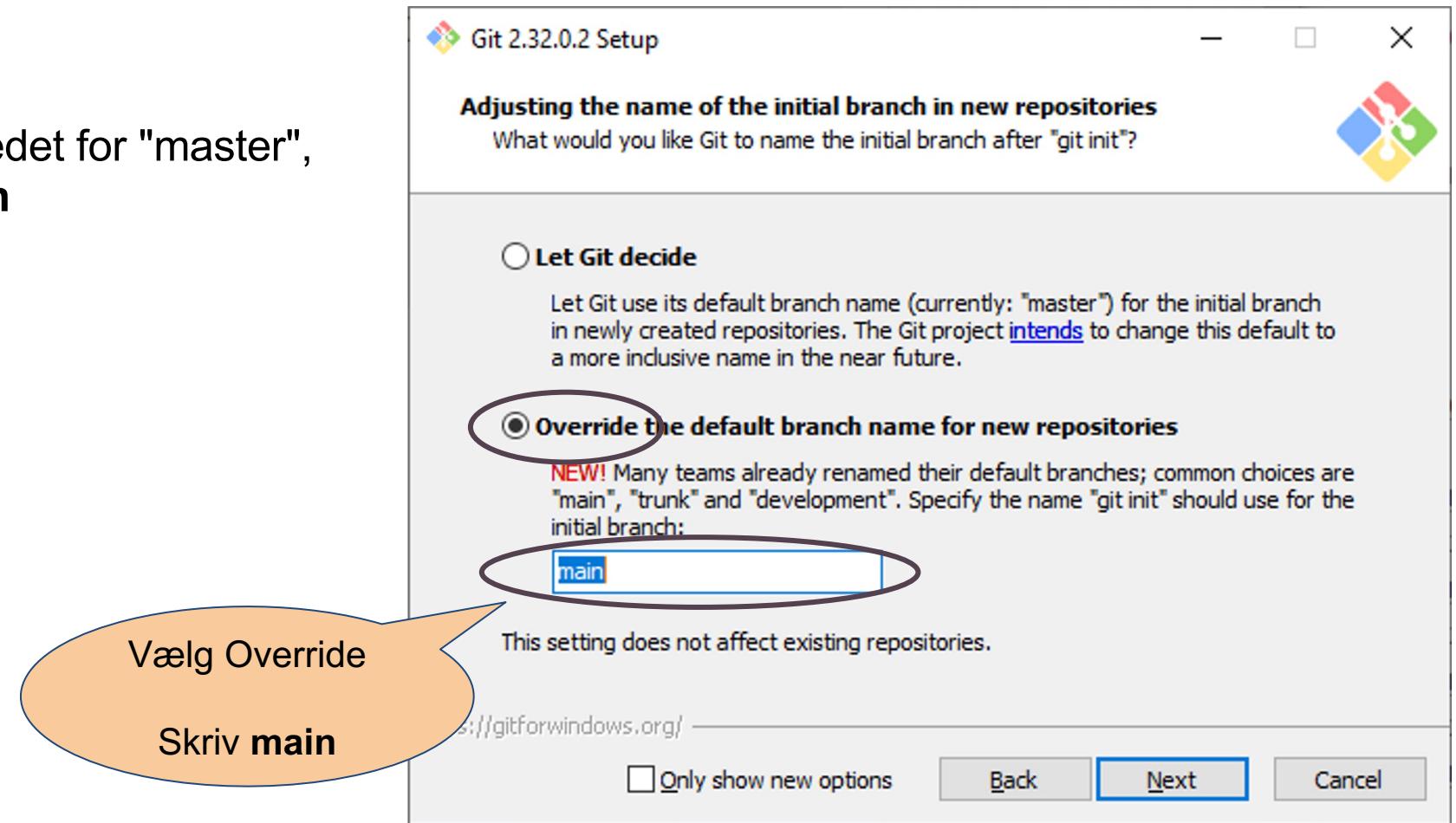
Alt efter hvad du har installeret på din maskine, får du forskellige muligheder.

Notepad er et glimrende valg!



Initial branch

På GitHub bruger de **main** i stedet for "master",
så vælg override, og skriv **main**



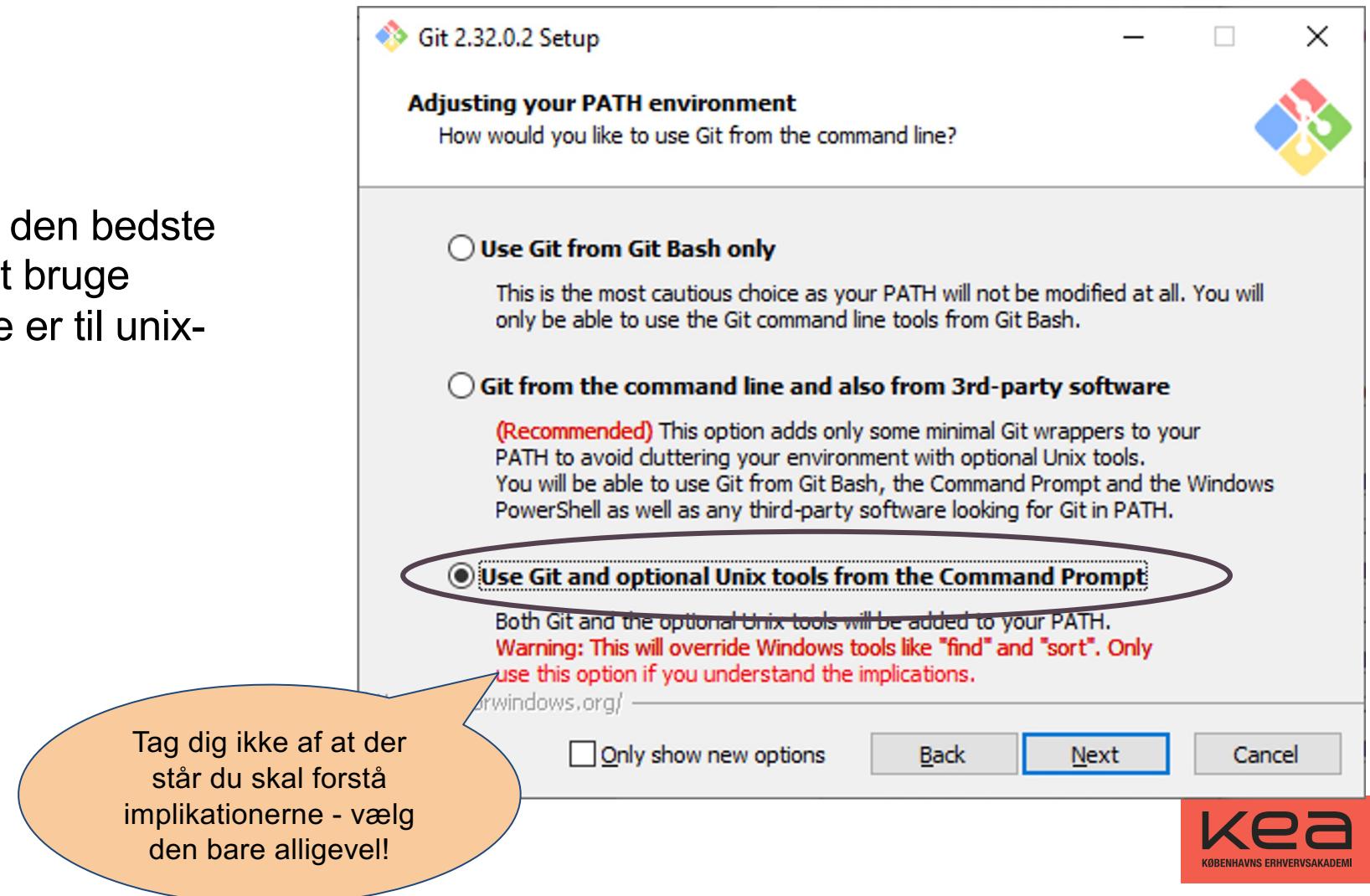
INSTALLER GIT (WINDOWS)

(5/15)

PATH

Vælg den nederste mulighed!

Selvom den advarer, så giver det den bedste oplevelse, og gør det nemmere at bruge vejledninger fra nettet, selv om de er til unix-systemer.



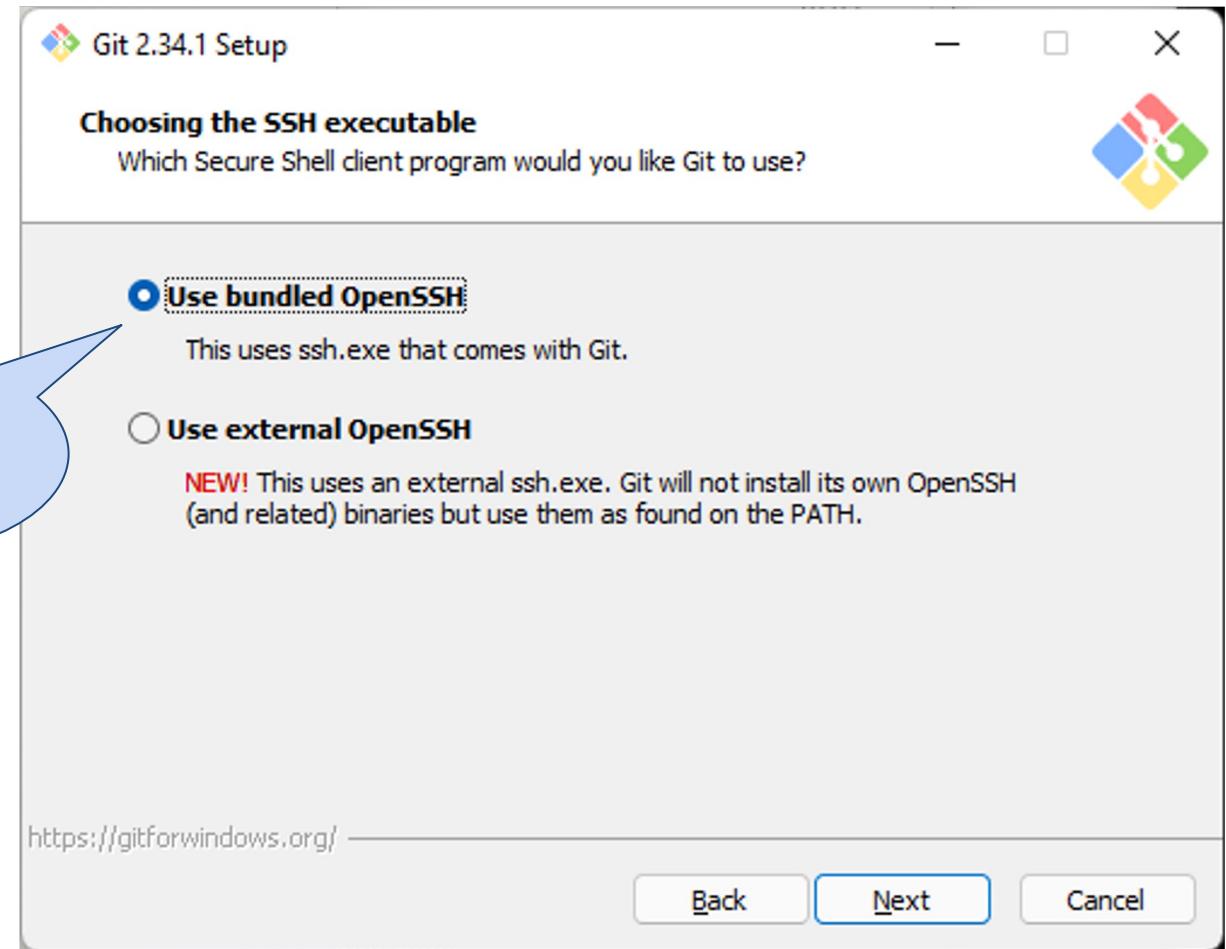
Tag dig ikke af at der
står du skal forstå
implikationerne - vælg
den bare alligevel!

SSH

Denne side får du muligvis ikke - det er kun hvis man har installeret alternative SSH programmer.

Hvis den spørger, så vælg
bundled OpenSSH

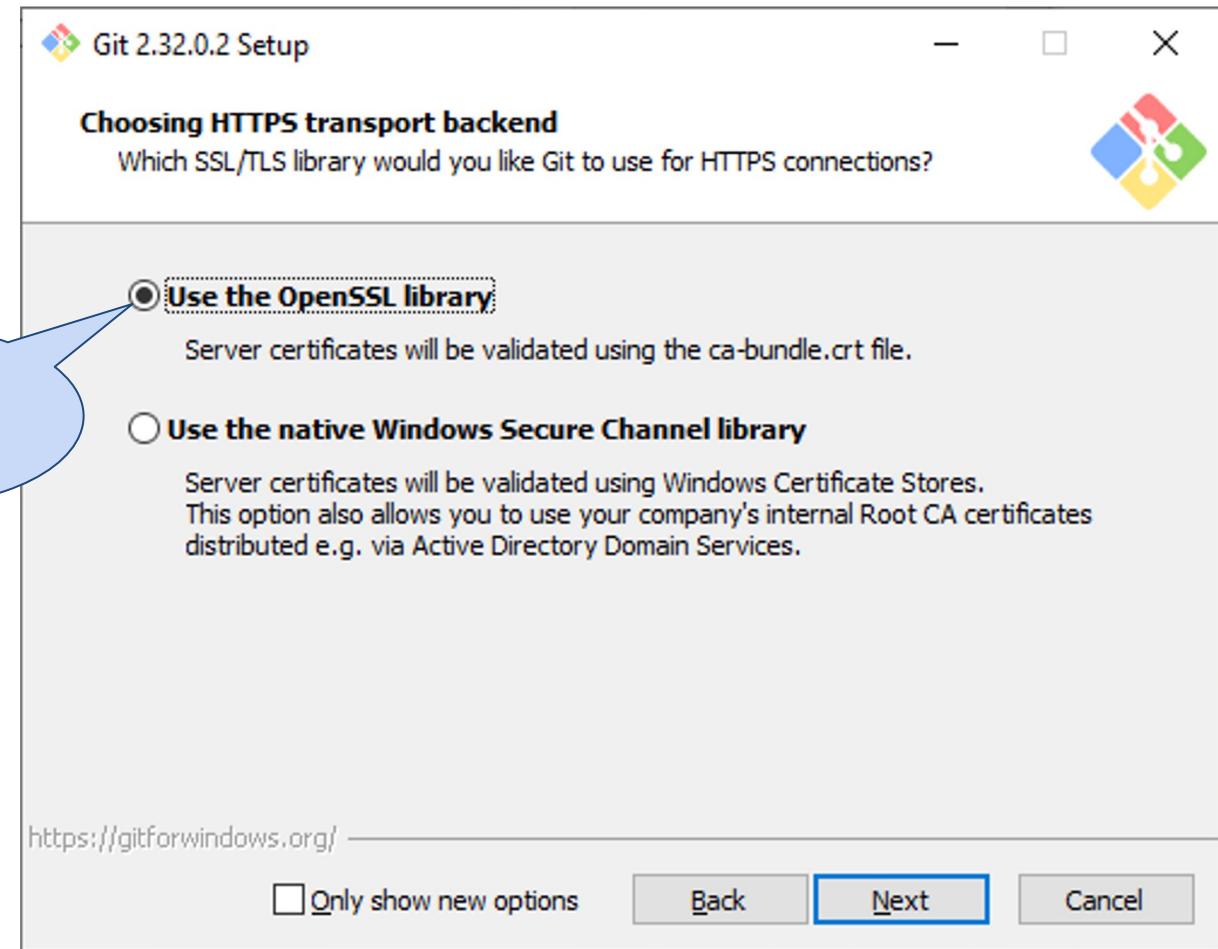
vælg
bundled
OpenSSH



OpenSSL

Du får næppe brug for dette, men vælg OpenSSL, så undgår du konflikter med andre programmer.

vælg
OpenSSL

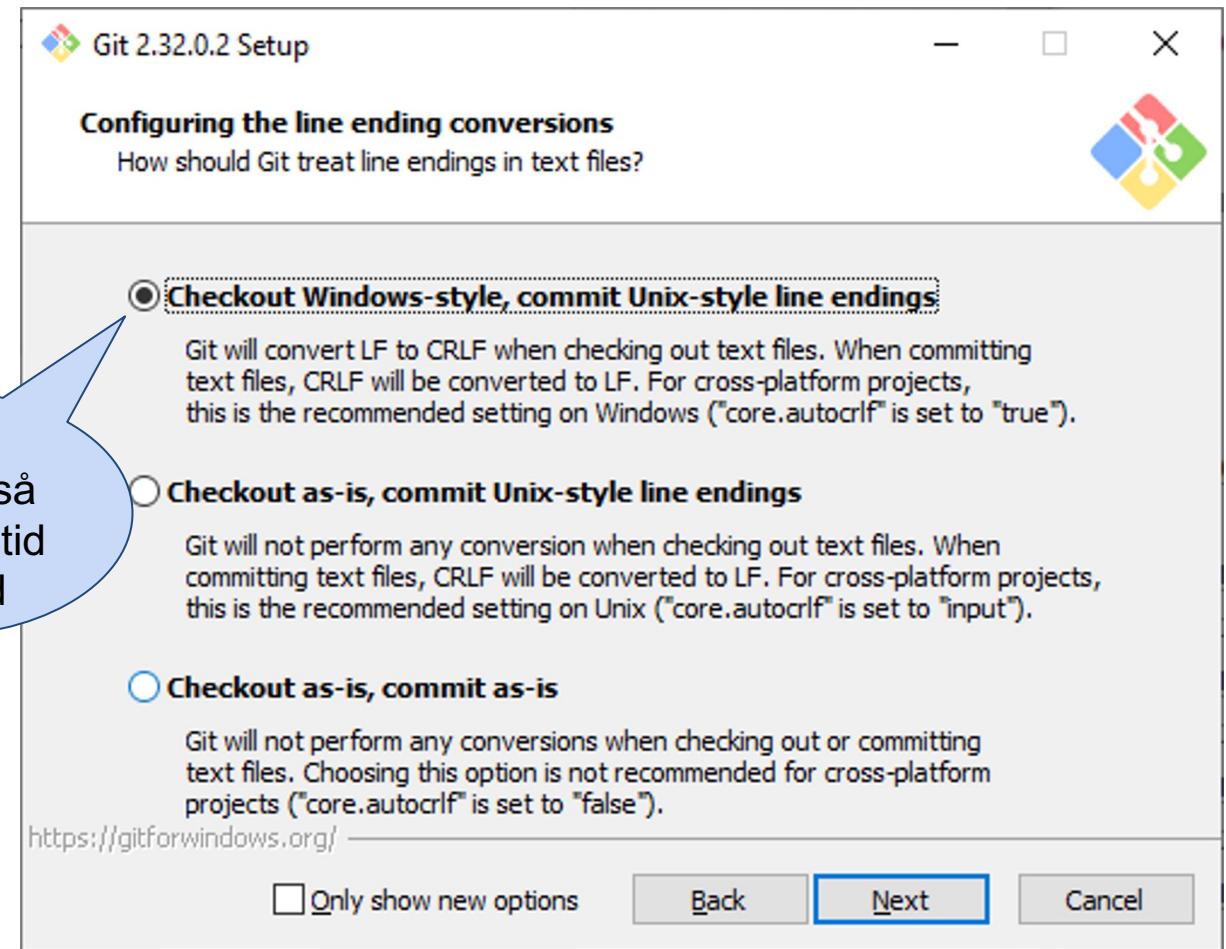


Line endings

Windows indsætter et ekstra linjeskift efter hver linje, så deler du kode med Mac eller Linux-brugere, får de en masse tomme linjer.

Vælg derfor at Git skal fjerne de ekstra linjer.

Vælg
denne - så
ser det altid
godt ud



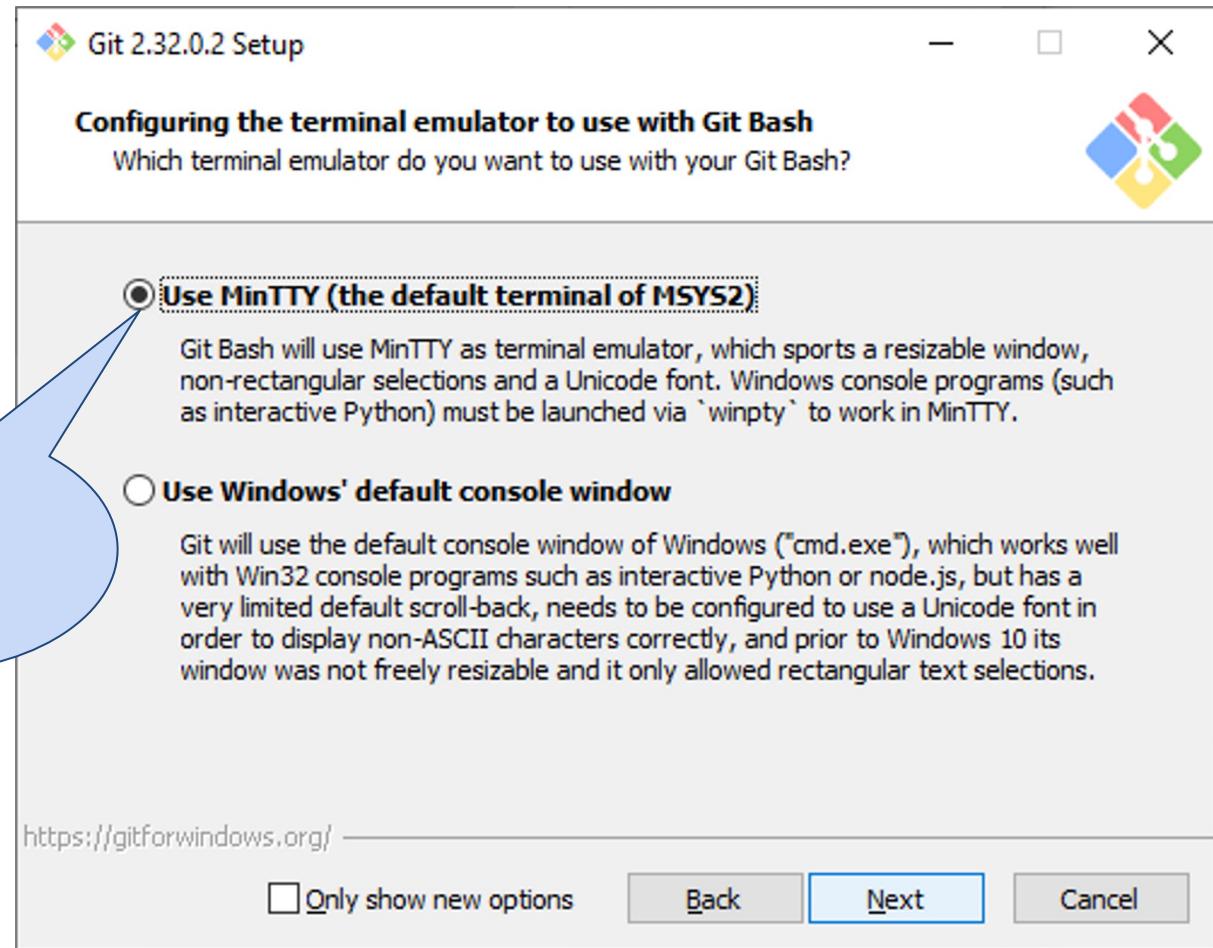
INSTALLER GIT (WINDOWS)

(9/15)

MinTTY

Medmindre du har konfigurereret din cmd.exe til at se rigtig lækker ud, så vælg MinTTY her - du kommer sikkert alligevel ikke til at se den ...

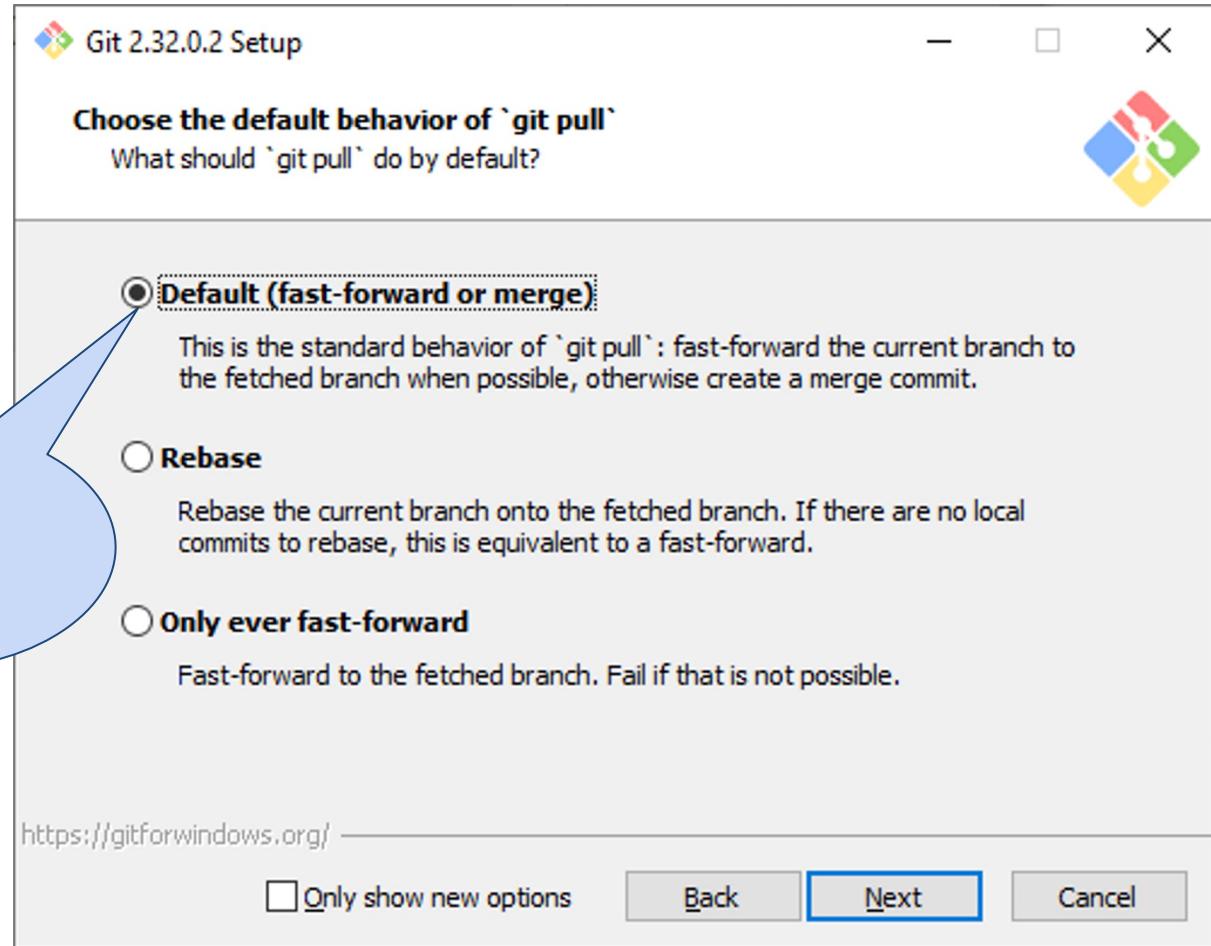
Vælg
MinTTY



git pull

Vælg default indstillingen - alt andet vil forvirre.

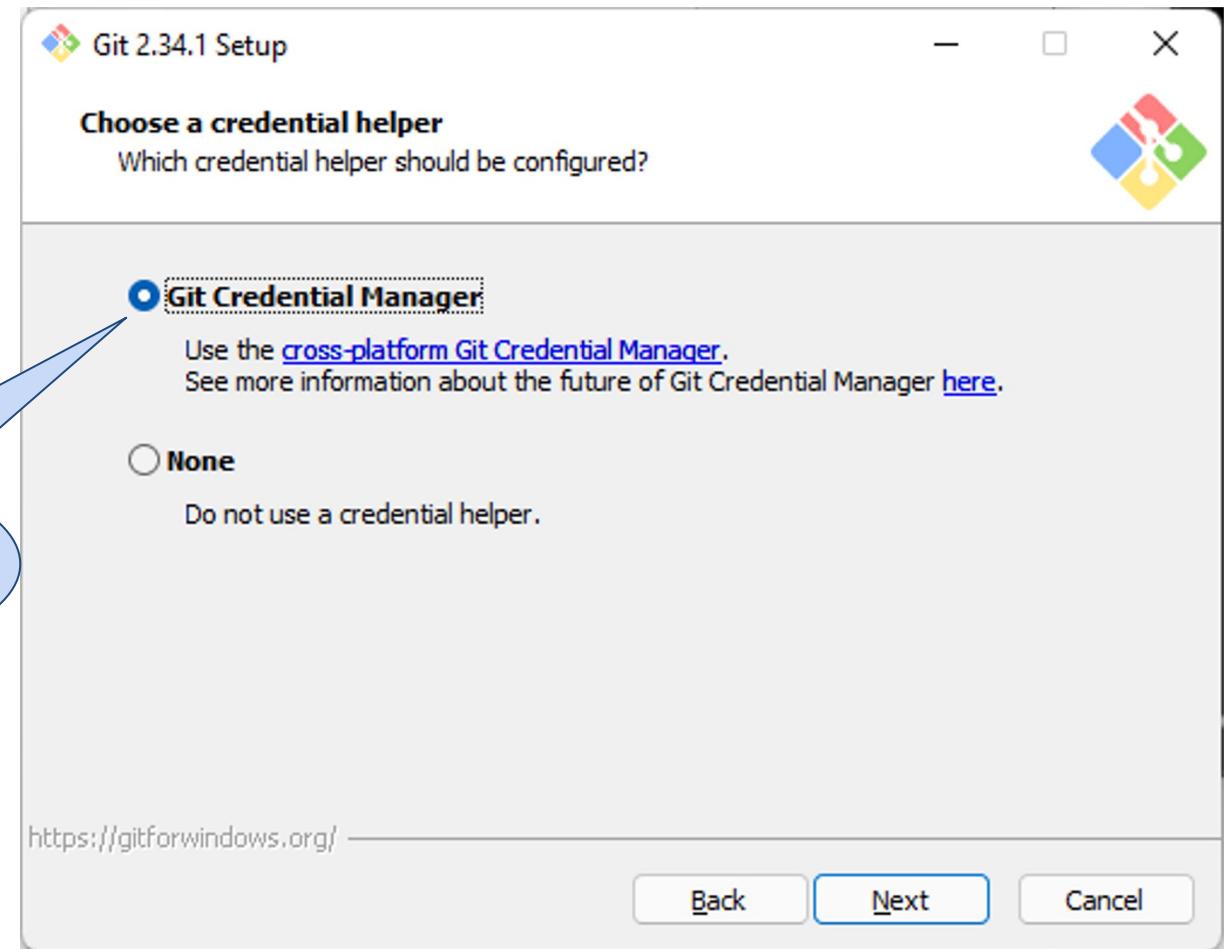
Vælg
Default



Git Credential Manager

Sørg for at bruge en credential helper - ellers bliver du bedt om passwords / access tokens hele tiden.

Sørg for at den
er valgt her



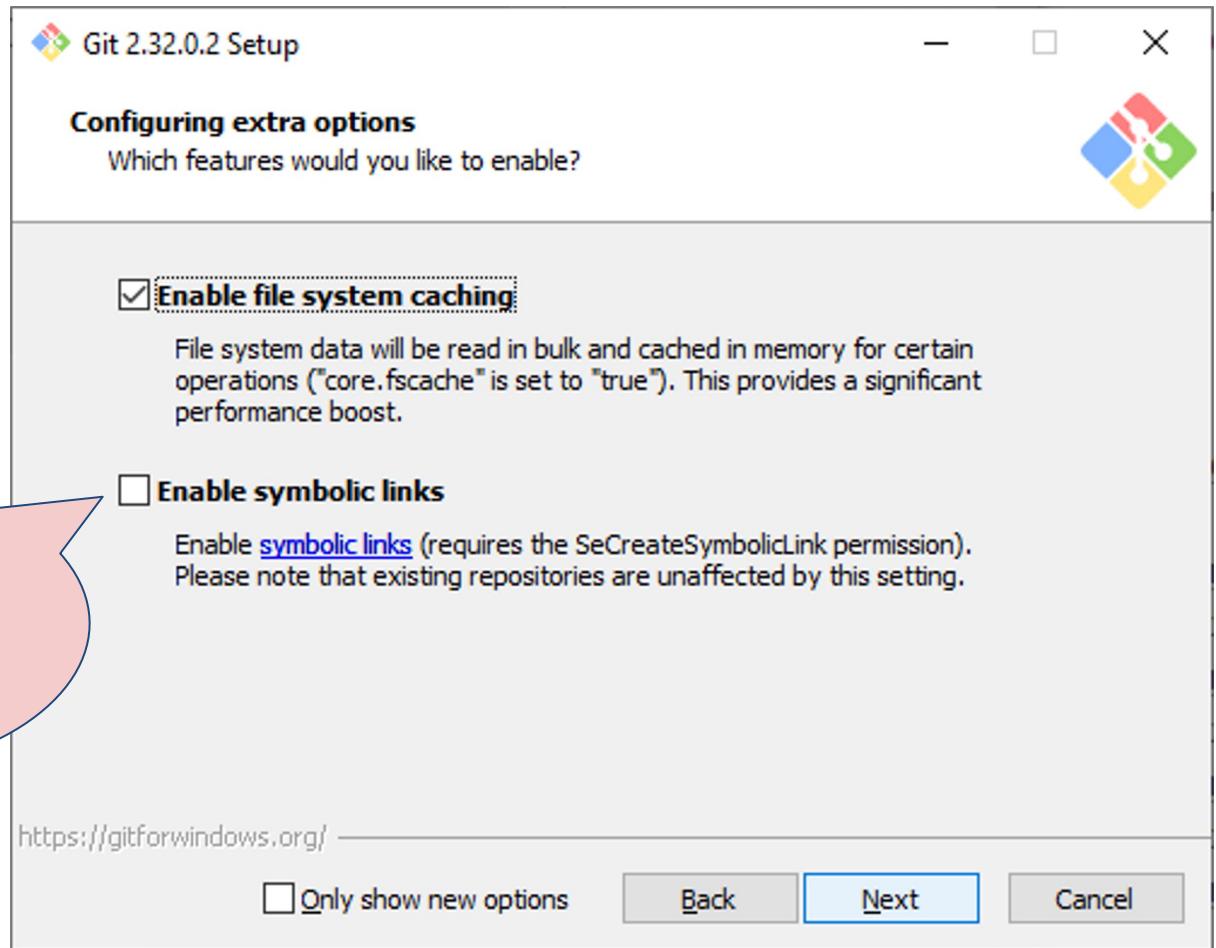
INSTALLER GIT (WINDOWS)

(12/15)

Extra options

Slå caching til, men symbolic links fra

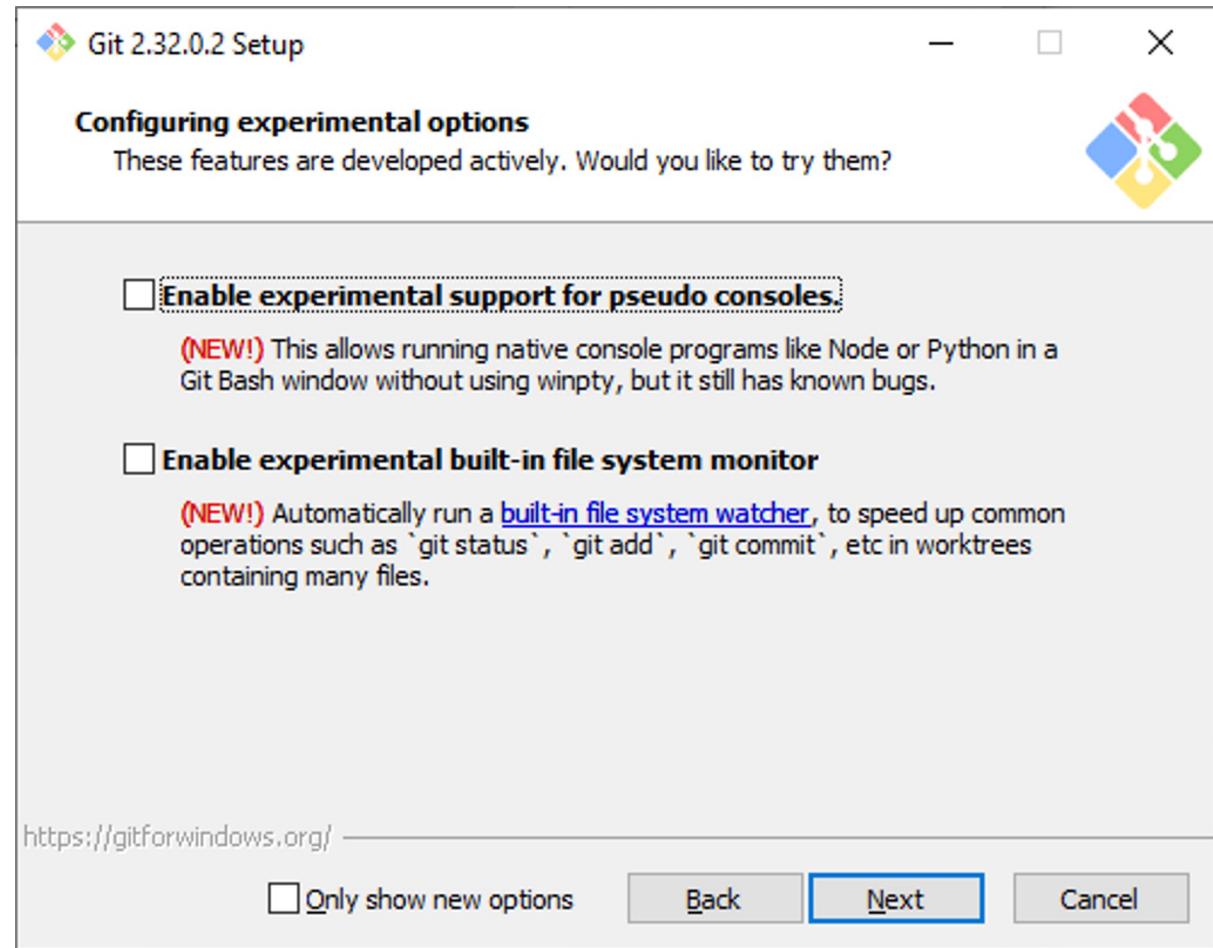
undlad symbolic
links - de ender
kun med at
forvirre os



Experimental options

Du er fri til at eksperimentere med disse.

Jeg ved ikke hvad de gør :)

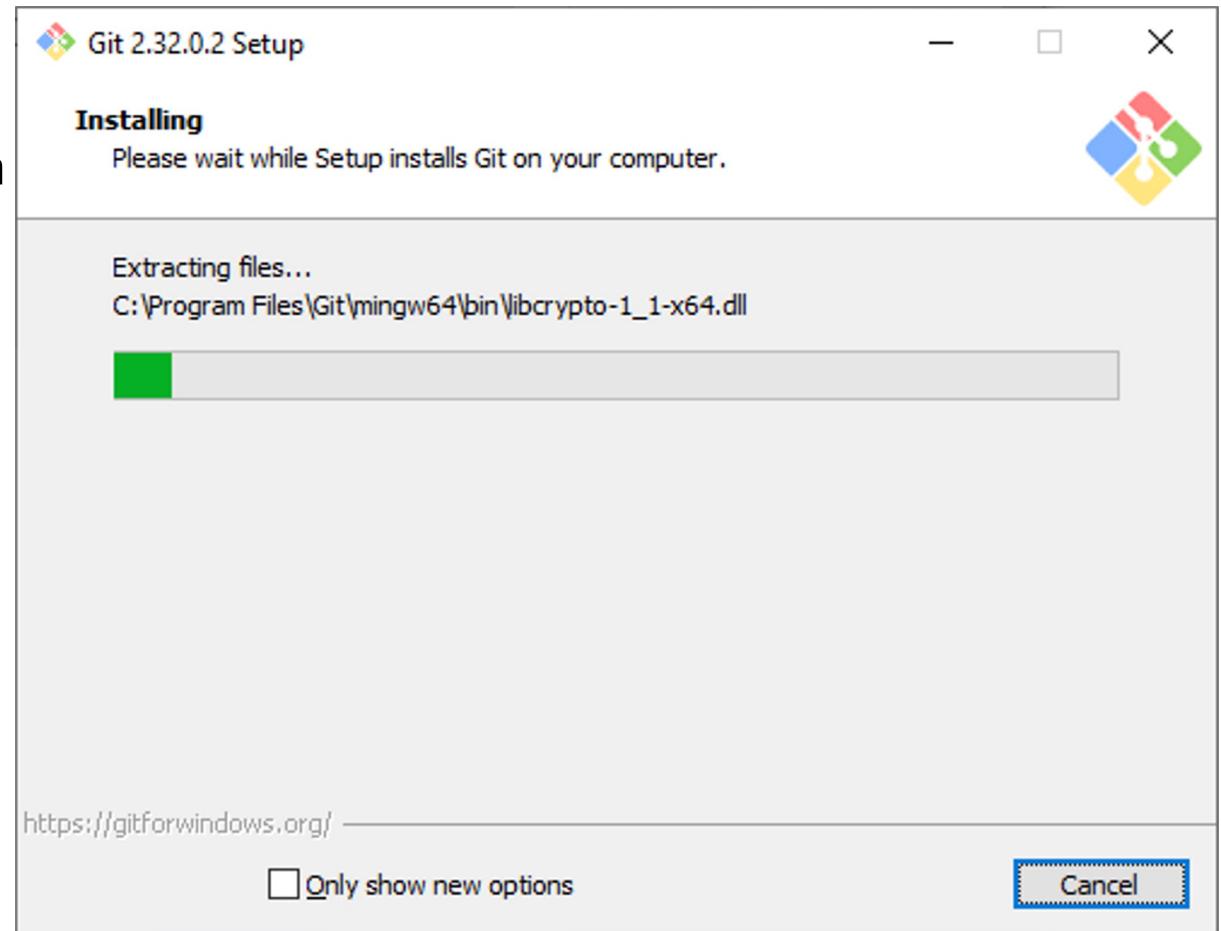


INSTALLER GIT (WINDOWS)

(14/15)

Så er du igennem alle valg, og git bliver installeret.

Det er blot at læne sig tilbage, og se progress-baren blive fyldt ud.



INSTALLER GIT (WINDOWS)

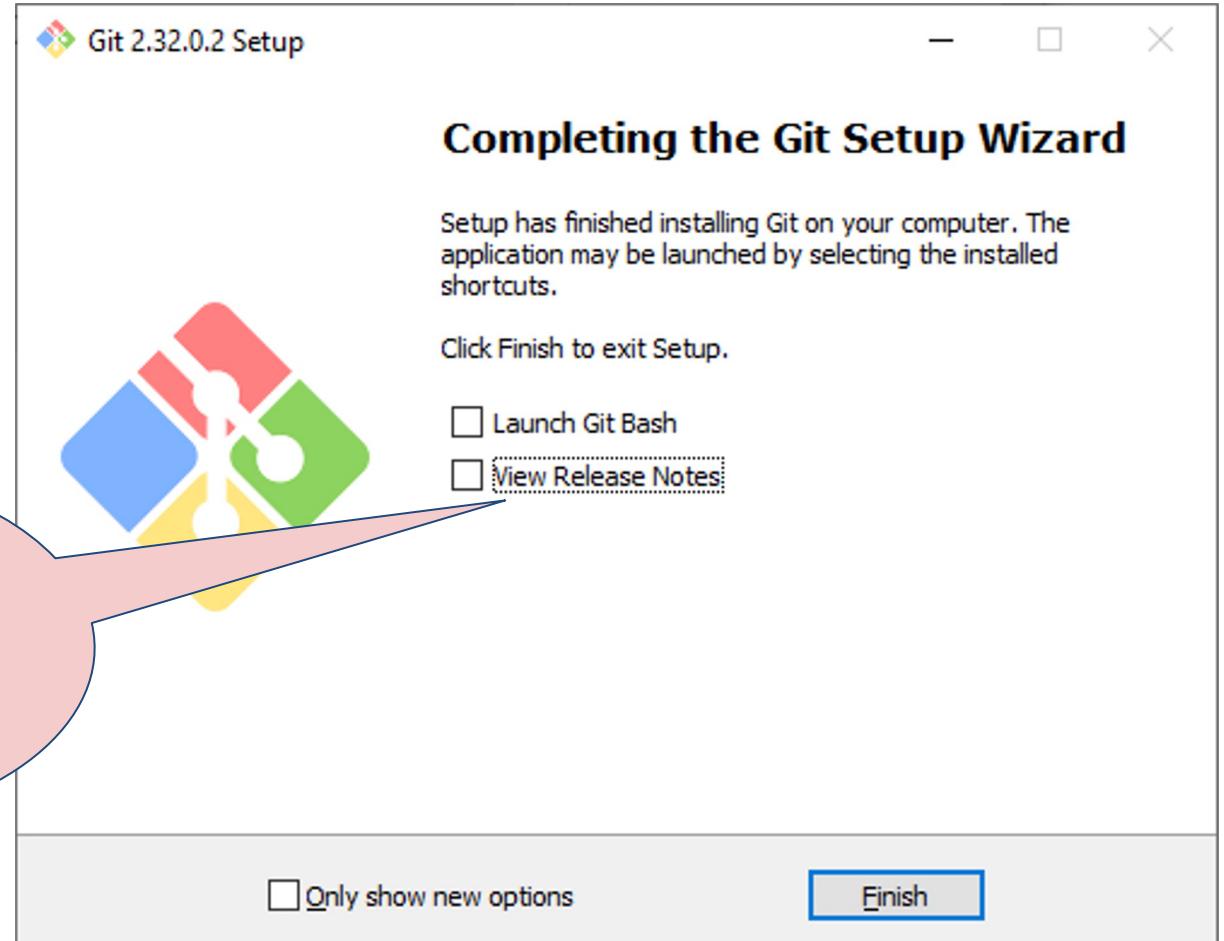
(15/15)

Complete

Så er det blot at undgå at læse release notes, og klikke Finish.

Så er git installeret!

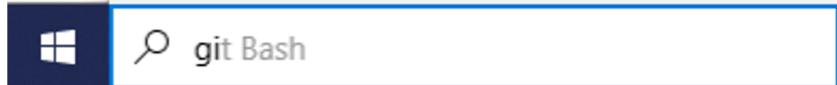
Slå "View Release Notes" fra - dem gider du alligevel ikke læse!



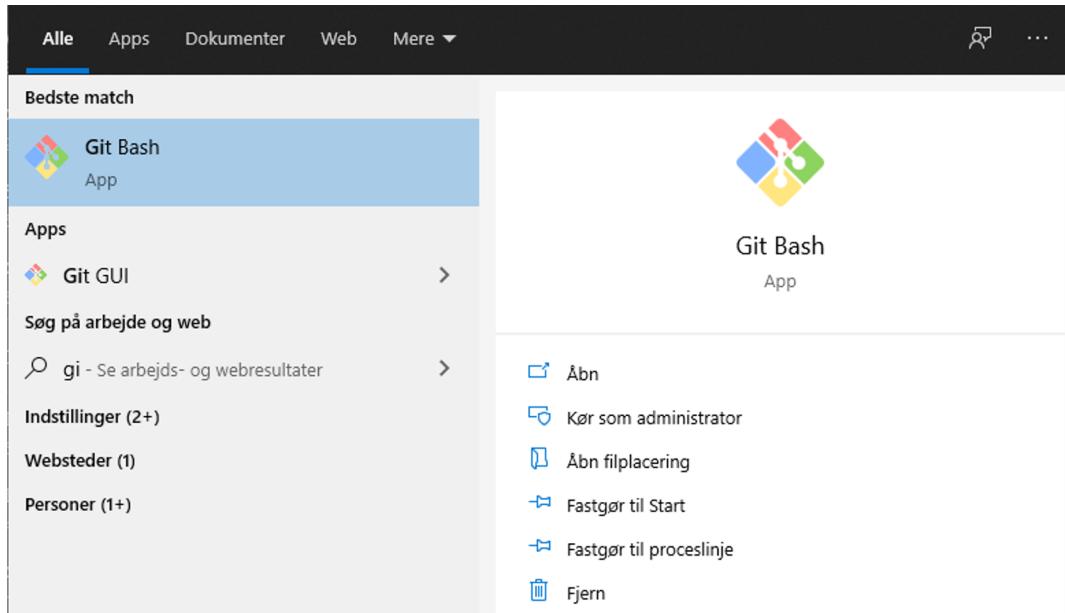
TEST INSTALLATIONEN AF GIT

Åben Git Bash:

Tryk på Windows-tasten for at aktivere søgerfeltet, og søg efter git bash

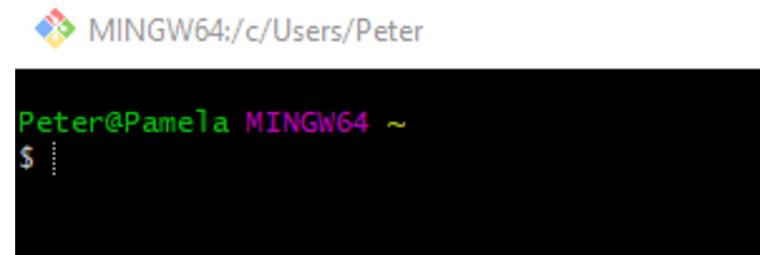


Åbn programmet



TEST INSTALLATIONEN AF GIT

Git Bash vinduet vil se nogenlunde således ud:

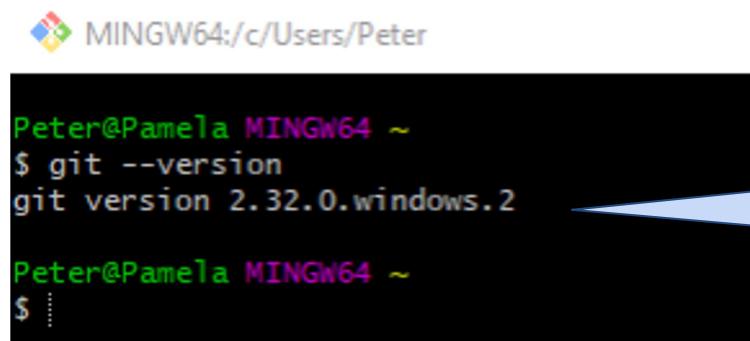


```
MINGW64:/c/Users/Peter  
Peter@Pamela MINGW64 ~  
$ ..
```

Men med dit navn og din computers navn i stedet.

Skriv `git --version` og tryk enter

Det skulle gerne resultere i et svar noget lignende dette:



```
MINGW64:/c/Users/Peter  
Peter@Pamela MINGW64 ~  
$ git --version  
git version 2.32.0.windows.2  
Peter@Pamela MINGW64 ~  
$ ..
```

Tjek at
versionsnummeret
svarer til den version
du downloadede

GITHUB

EFTER EN LILLE ANIMATION ÅBNER DIN GITHUB SIDE SIG

The screenshot shows the GitHub homepage with several interactive elements highlighted by hand-drawn style annotations:

- An orange callout bubble on the left side points to the **Create repository** button in the top navigation bar.
- A blue callout bubble on the right side points to the **Save the Date!** announcement for GitHub Universe.
- The main content area features a central modal window with the heading "Learn Git and GitHub without any code!" and a sub-section "Introduce yourself".
- At the bottom of the main content area, there are two buttons: "Dismiss this" and "Continue".
- The top navigation bar includes links for **Pull requests**, **Issues**, **Marketplace**, and **Explore**.
- The right side of the page shows a sidebar with sections for **Recent activity** and **All activity**.

LAV ET NYT REPOSITORY - BARE TIL TEST

Et "repository" er der hvor git (og GitHub) samler alle filer til et projekt.

Lav et lille test-projekt til HelloWorld.

Brug standard-indstillingerne: Public, og alt andet slået fra

Create a new repository
A repository contains all project files, including the revision
[Import a repository.](#)

Owner *  petertest21 / Repository name * 

Great repository names are HelloWorld is available. Need inspiration? How about [ubiquitous-train?](#)

Description (optional)

 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.

Initialize this repository with:
Skip this step if you're importing an existing repository.

Add a README file
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

[Create repository](#)

Tryk Create repository

Du bestemmer selv navnet her

men HelloWorld er et fint valg

sørg for at alt dette er slået fra

DET GIVER EN SIDE MED EN MASSE INFORMATIONER

Det ser måske lidt voldsomt ud - men det eneste vi skal bruge er linket der står i den øverste blå boks.

løvrigt samme link som til siden selv!

Lad siden her stå åben i din browser, imens du åbner IntelliJ

The screenshot shows the GitHub 'Quick setup' page for a new repository named 'HelloWorld'. It includes:

- A header with 'Set up in Desktop' (selected), 'HTTPS', 'SSH', and a URL field containing <https://github.com/petertest21>HelloWorld.git>.
- Text: 'Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a `README`, `LICENSE`, and `.gitignore`'.
- A section titled '...or create a new repository on the command line' with the following code:

```
echo "# HelloWorld" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/petertest21>HelloWorld.git
git push -u origin main
```
- A section titled '...or push an existing repository from the command line' with the following code:

```
git remote add origin https://github.com/petertest21>HelloWorld.git
git branch -M main
git push -u origin main
```
- A section titled '...or import code from another repository' with the note: 'You can initialize this repository with code from a Subversion, Mercurial, or TFS project.' and a 'Import code' button.

💡 ProTip! Use the URL for this page when adding GitHub as a remote.

FORBIND GIT OG GITHUB

GIT SKAL OGSÅ KENDE DIT BRUGERNAVN

Git er blot et program der kører på din maskine, og har som sådan ikke noget som helst at gøre med GitHub – men for at undgå forvirring vil vi sætte Git op til at bruge samme brugernavn og email som GitHub.

Åbn en terminal hvis du er på Mac, eller Git-Bash hvis du er på Windows, og skriv:

```
git config --global user.name "brugernavn"
```

tryk enter og skriv

```
git config --global user.email "din@email.adresse"
```

og tryk enter igen – så ved Git hvem du er!

Brug dit GitHub
brugernavn – så
er det nemmere

skal være den
email du brugte
på GitHub!

FORBIND INTELLIJ OG GITHUB

ÅBN INTELLIJ

Find dit oprindelige HelloWorld projekt - eller lav et nyt, hvis det er blevet væk.

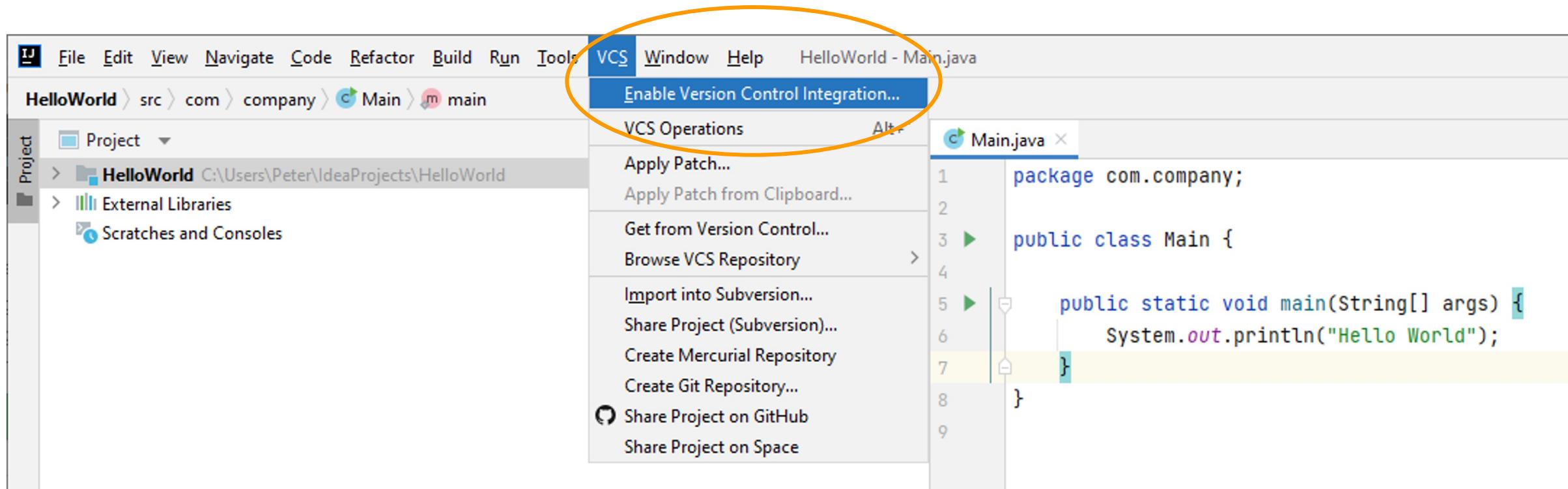
I det følgende får du sat IntelliJ op til at bruge et lokalt git repository til at gemme projektet - og forbundet det lokale repository med det du lige har lavet på GitHub.

Tag dig ikke af at koncepterne og begreberne er lidt mystiske - tænk bare på at det hele går ud på at få Java source filerne fra din maskine, op på GitHub siden, så du kan dele dem med resten af verden.

Og her i denne guide skal du bare afprøve det - i selve undervisningen begynder vi så at arbejde med det, og forstå det lidt mere.

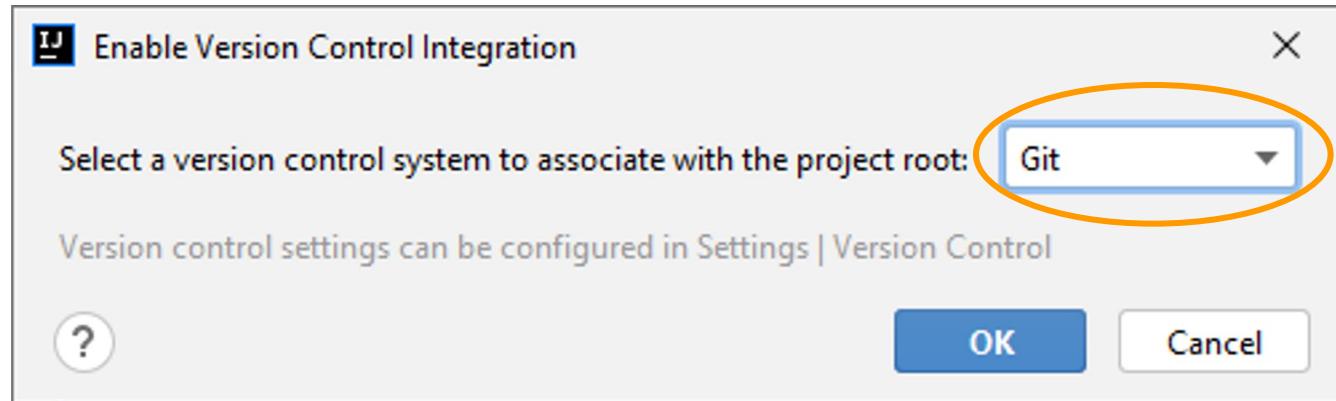
VÆLG VCS -> ENABLE VERSION CONTROL INTEGRATION

Det sætter dit projekt op til at bruge versionsstyring - IntelliJ kan arbejde med mere end git, så vi skal lige fortælle hvilken type det er vi bruger.



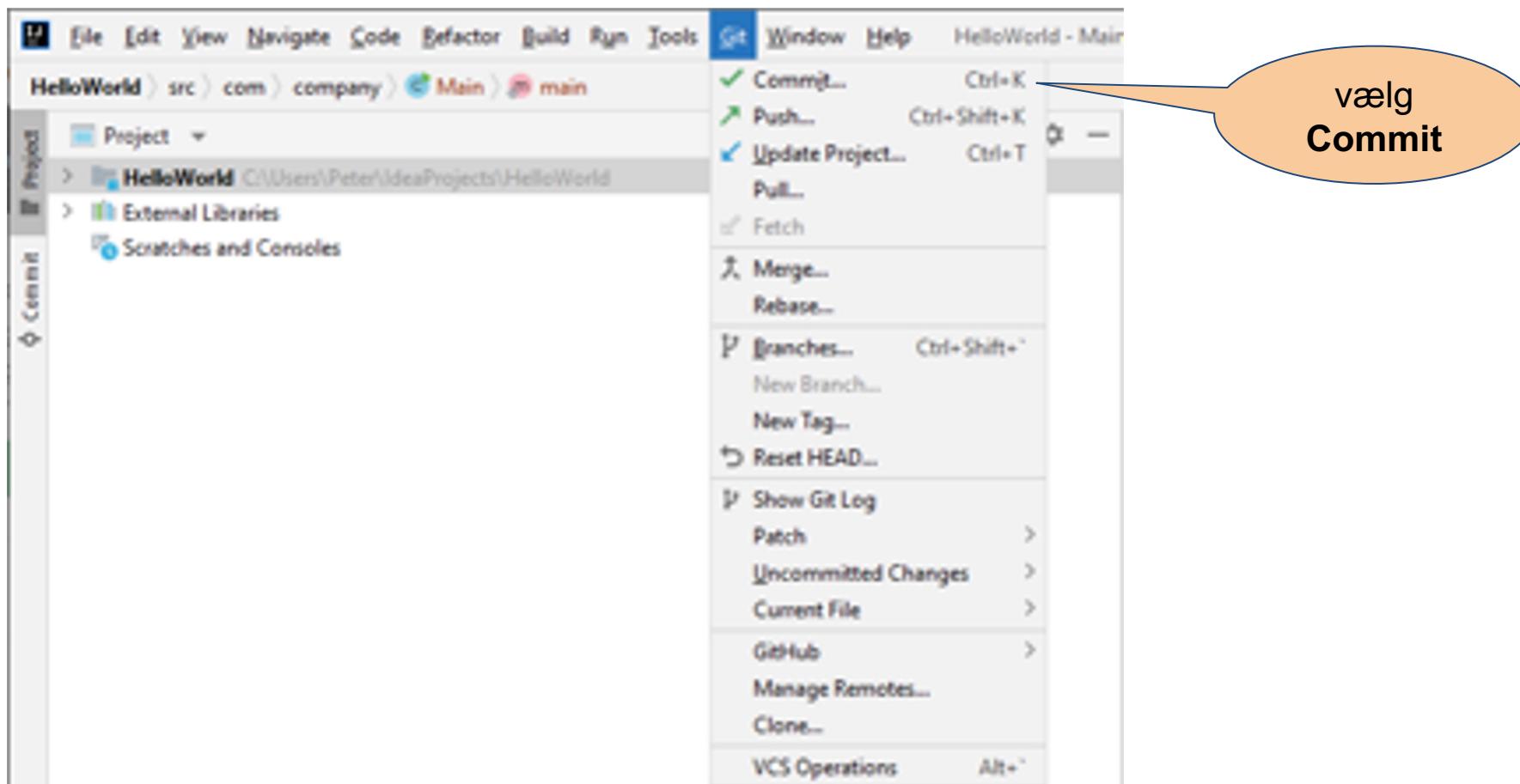
ENABLE VERSION CONTROL INTEGRATION → VÆLG GIT

Enable Version Control Integration åbner et lille vindue, hvor det spørger hvilken slags - og her vælger du Git



NU ER DER EN GIT MENU I STEDET FOR VCS

Dit projekt er nu sat op til at bruge Git, og i menuen har du alle ønskelige Git-funktioner lige ved hånden.



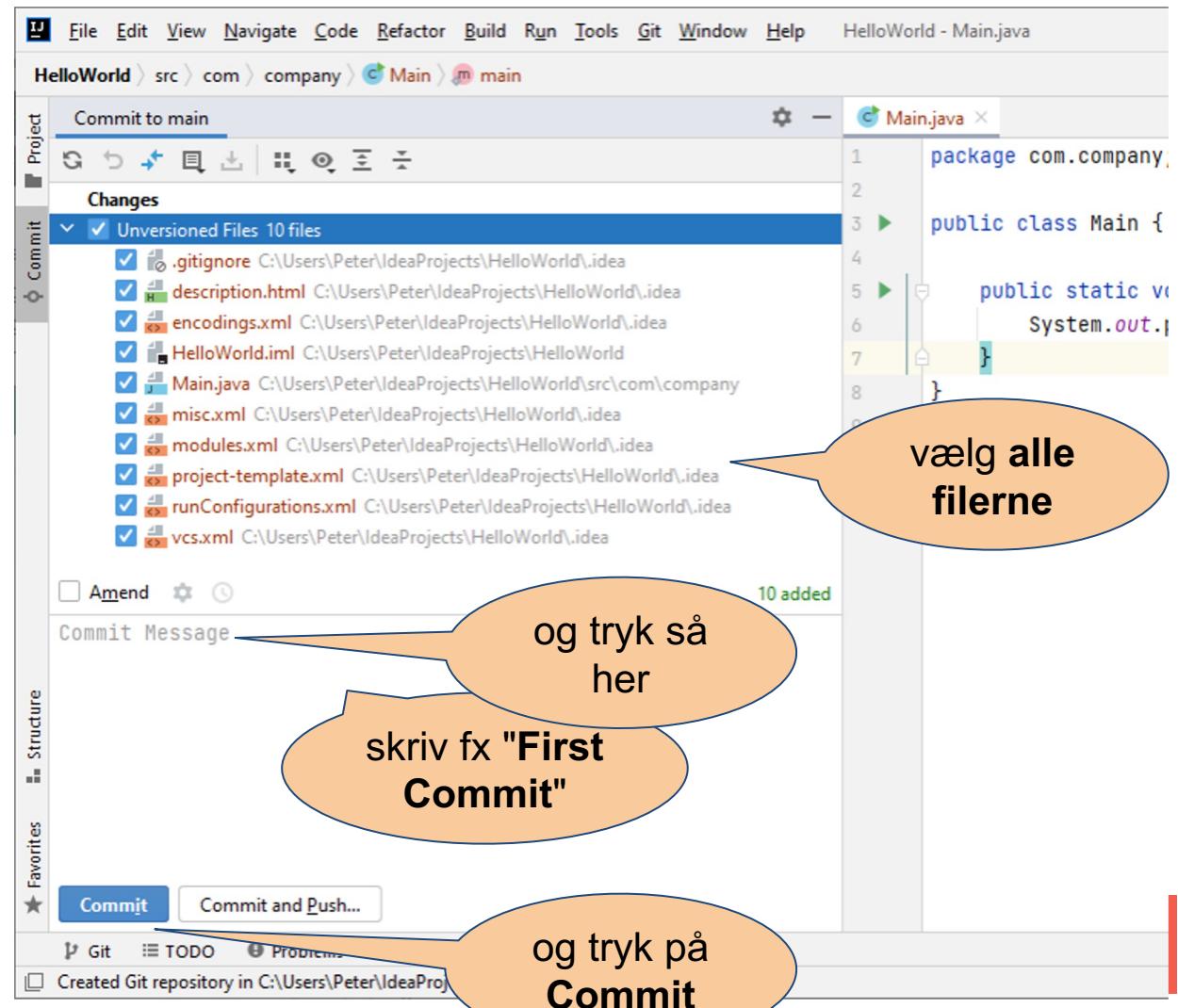
COMMIT ALLE FILER

Git holder kun styr på de filer vi beder den om - og vi beder den om det ved at "committe" filer.

Vælg alle filerne og klik i den nederste del af vinduet hvor der står "Commit Message"

Her skriver du en kort besked om hvorfor du committede de filer. "First Commit" er som regel et godt bud.

Endelig trykker du på Commit knappen

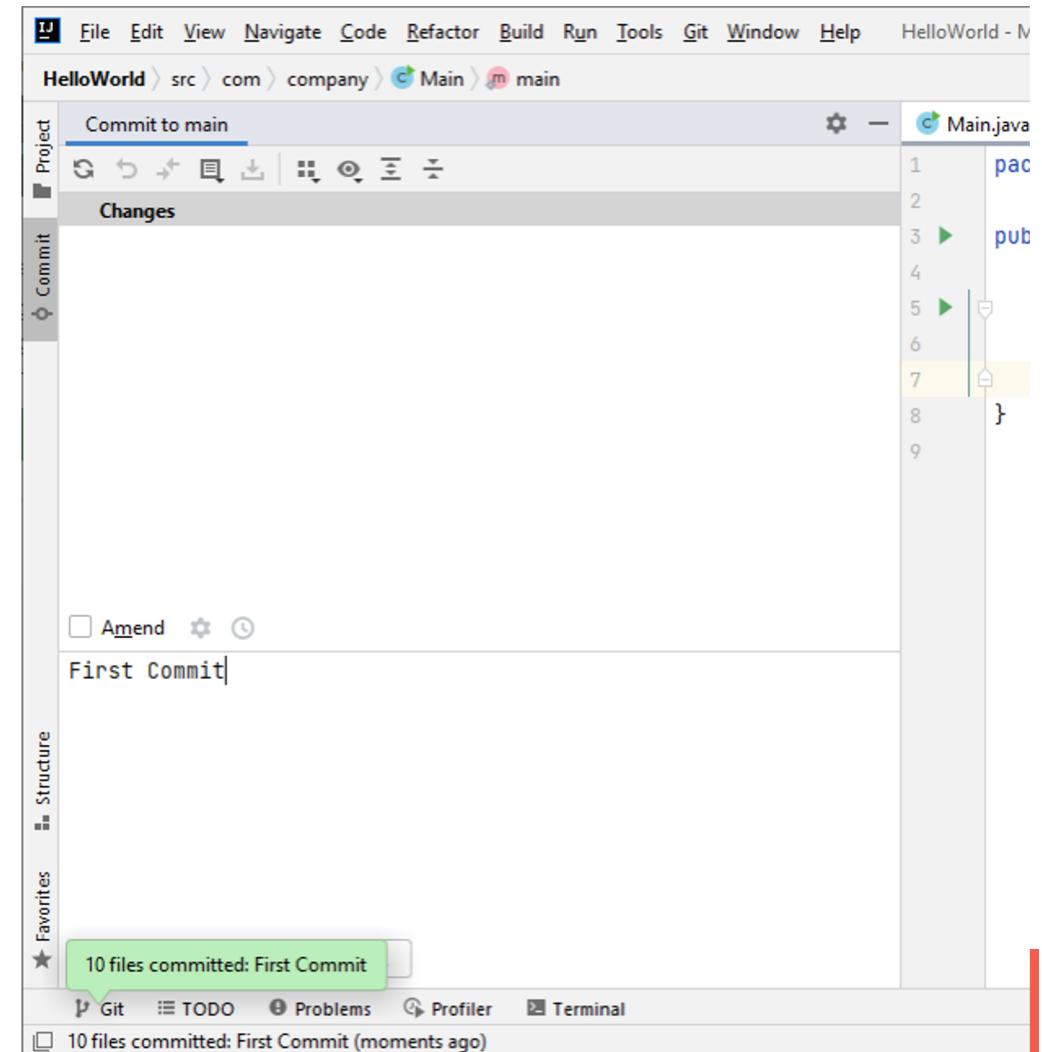


SÅ ER FILERNE COMMITTED

Det tager ganske kort tid, og du kan se at filerne forsvinder fra listen over ændrede filer, og der popper en lille taleboble op nederst, der fortæller at filerne blev committed.

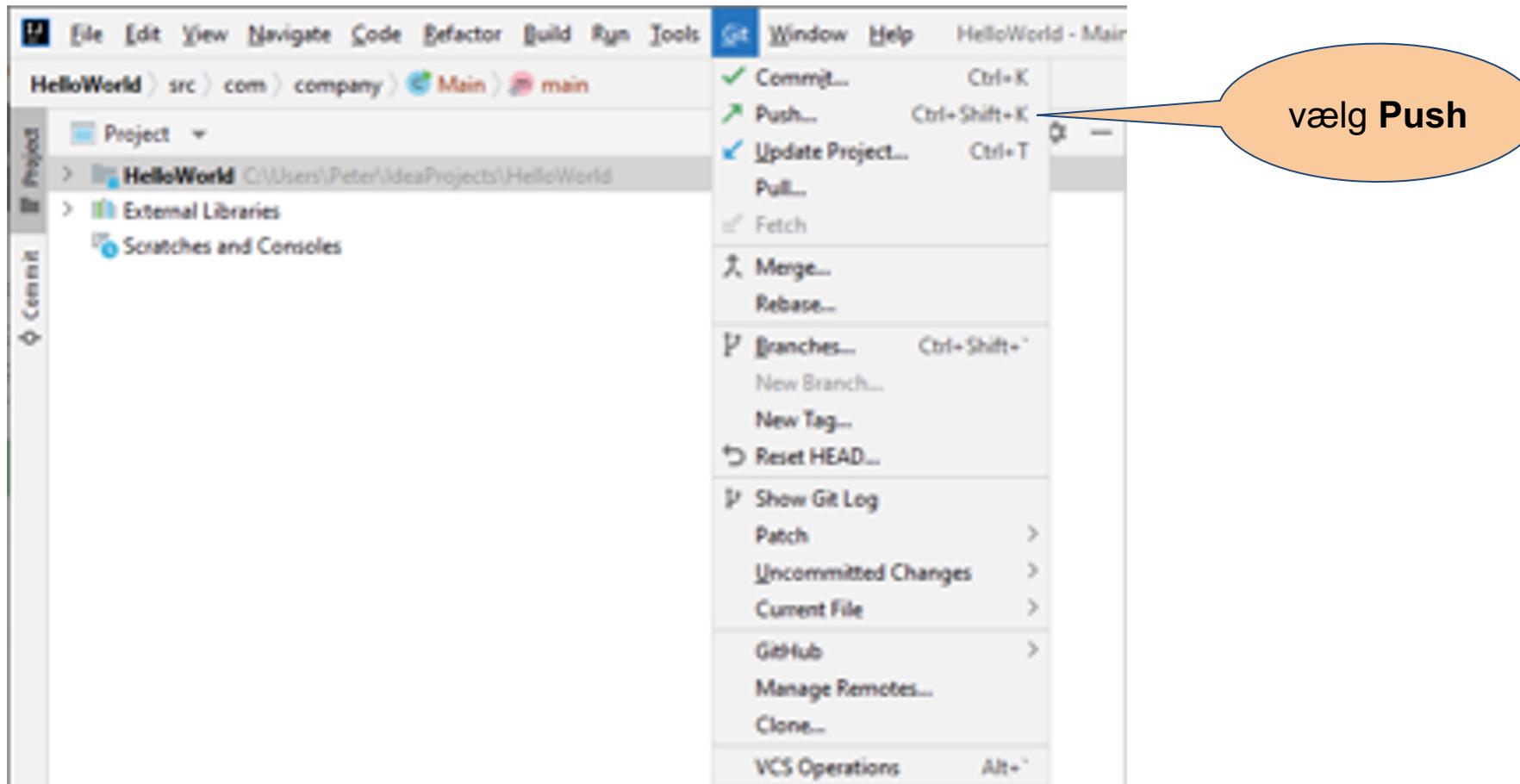
Nu ligger de i Git - og dette trin gentager man så normalt hver gang man vil gemme ændringerne i sin kode.

Men nu skal projektet op på GitHub



PUSH ÆNDRINGER TIL GITHUB

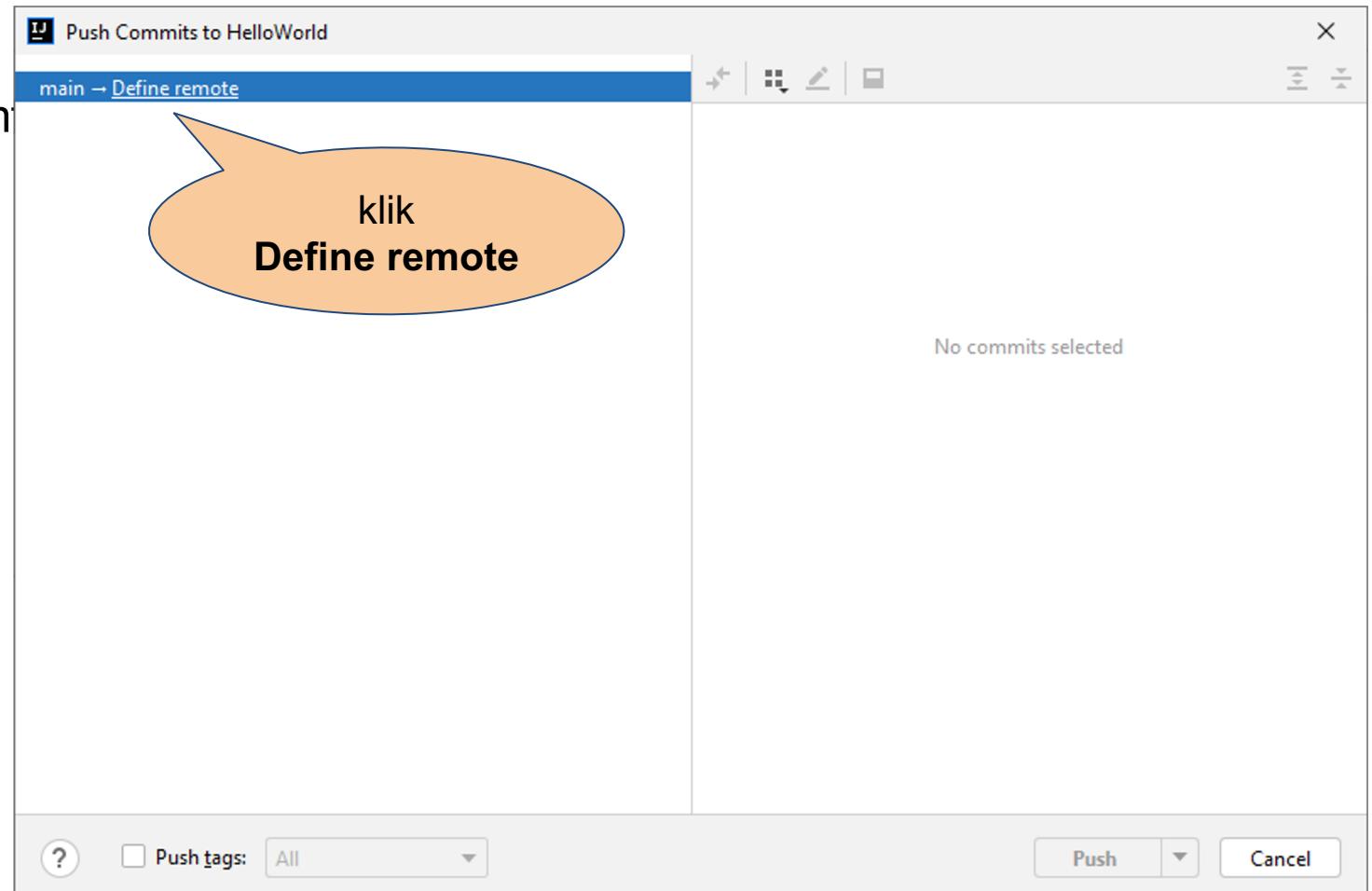
Du har allerede lavet et repository på GitHub - nu skal ændringerne du lige committede, pushes fra din maskine, og op til GitHub - så vælg Push i Git-menuen



DEFINE REMOTE

Push skal vide hvor der skal pushes hen, så den åbner et vindue der ser meget tom ud.

Klik på Define remote

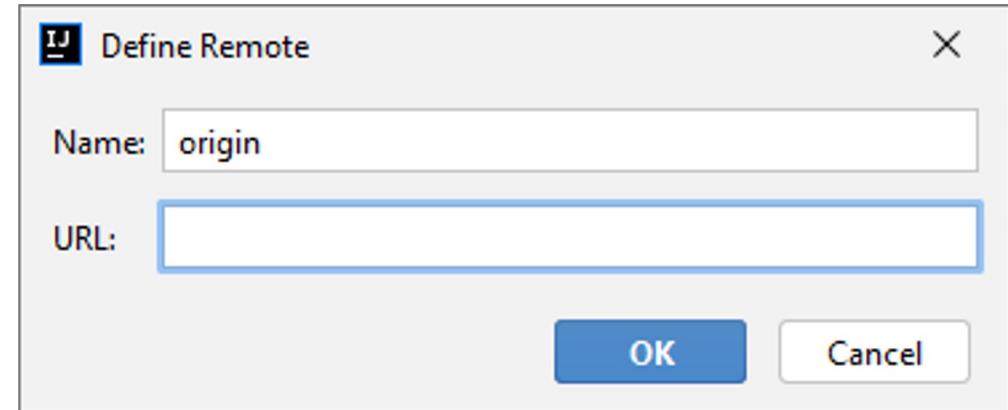


ANGIV URL'EN PÅ

Define Remote åbner et vindue der beder om en URL.

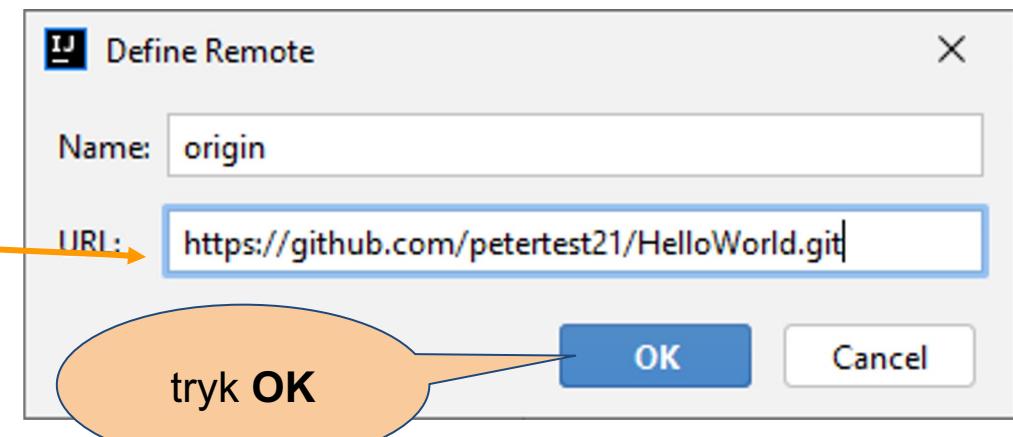
Det er den adresse du fik for snart længe siden, fra GitHub.

Kopier den fra browseren, og paste den ind i vinduet. (*Det er lige meget om der står .git til sidst eller ej*)



Quick setup — if you've done this kind of thing before
Set up in Desktop or HTTPS SSH https://github.com/peterest21>HelloWorld.git
Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

A yellow arrow points from the URL in the GitHub screenshot to the 'URL:' field in the 'Define Remote' dialog box below.

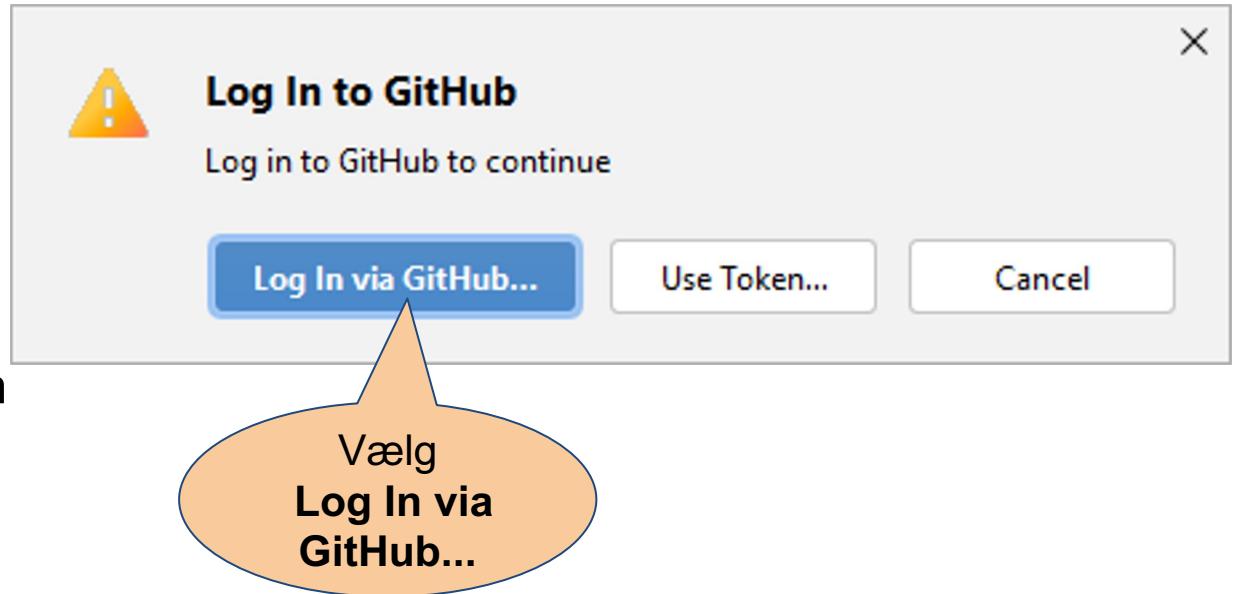


LOG IN VIA GITHUB...

Straks du har angivet remote URL, får du en "fejl".

IntelliJ skal nemlig have tilladelse til at tilgå din GitHub.

Det nemmeste er at vælge Log In via GitHub... det vil nemlig konfigurere din GitHub til at genkende din IntelliJ.



DET ÅBNER EN SIDE I DIN BROWSER

Sørg for at det er i den samme browser som du er logget ind på GitHub i! Hvis ikke, så bare kopier linket for siden over i den anden browser, og klik så på Authorize in GitHub



Det kan ske at denne side fejler, og du får en Error 403 som vist her til højre.

[Hvis du gør, så klik her for at se forslag til at rette op på den fejl.](#)

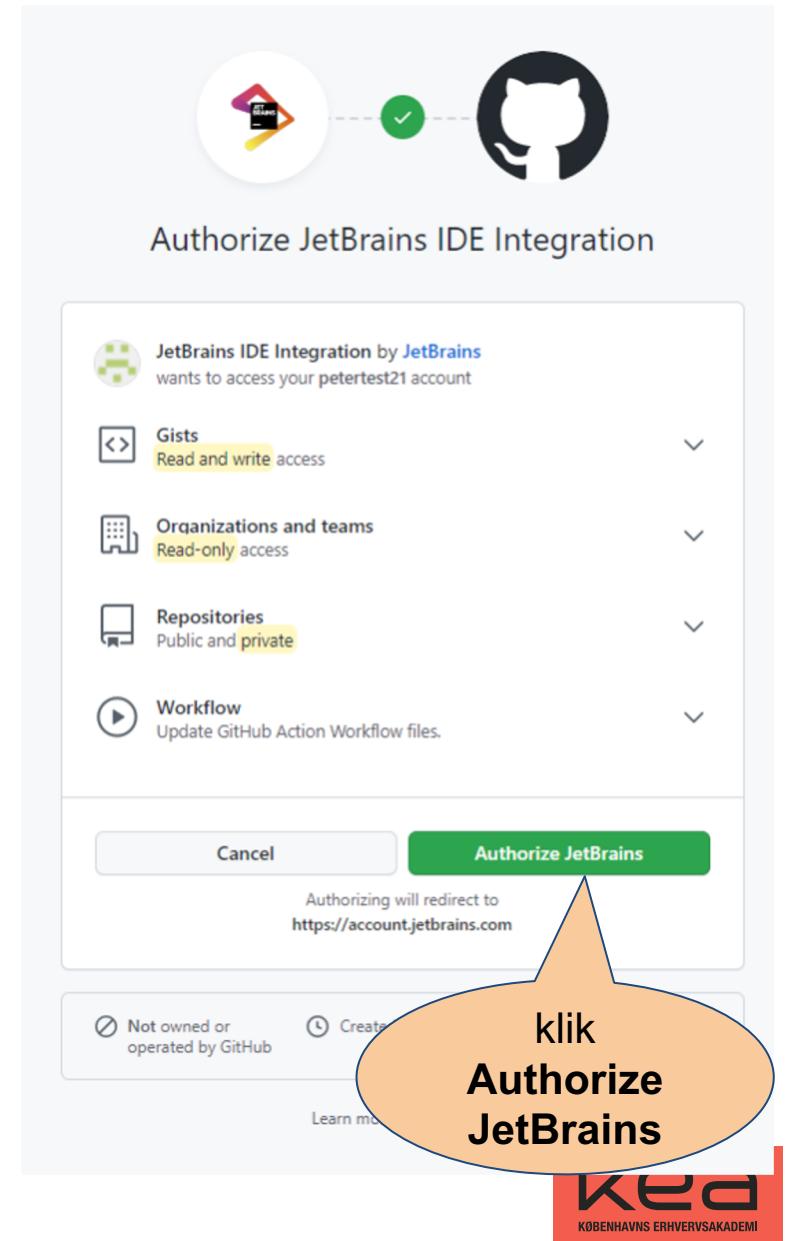


AUTHORIZE JETBRAINS IDE INTEGRATION

Det åbner så en side under din GitHub-konto, hvor du bliver bedt om at godkende at IntelliJ - faktisk JetBrains produkter - har adgang til din GitHub-konto.

Bare rolig - det er kun *din* udgave af IntelliJ der har adgang til *din* GitHub-konto

Så klik Authorize JetBrains



DET GÅR TILBAGE TIL DEN FOREGÅENDE SIDE

Du får så igen en forholdsvis tom side fra JetBrains som bekræftelse

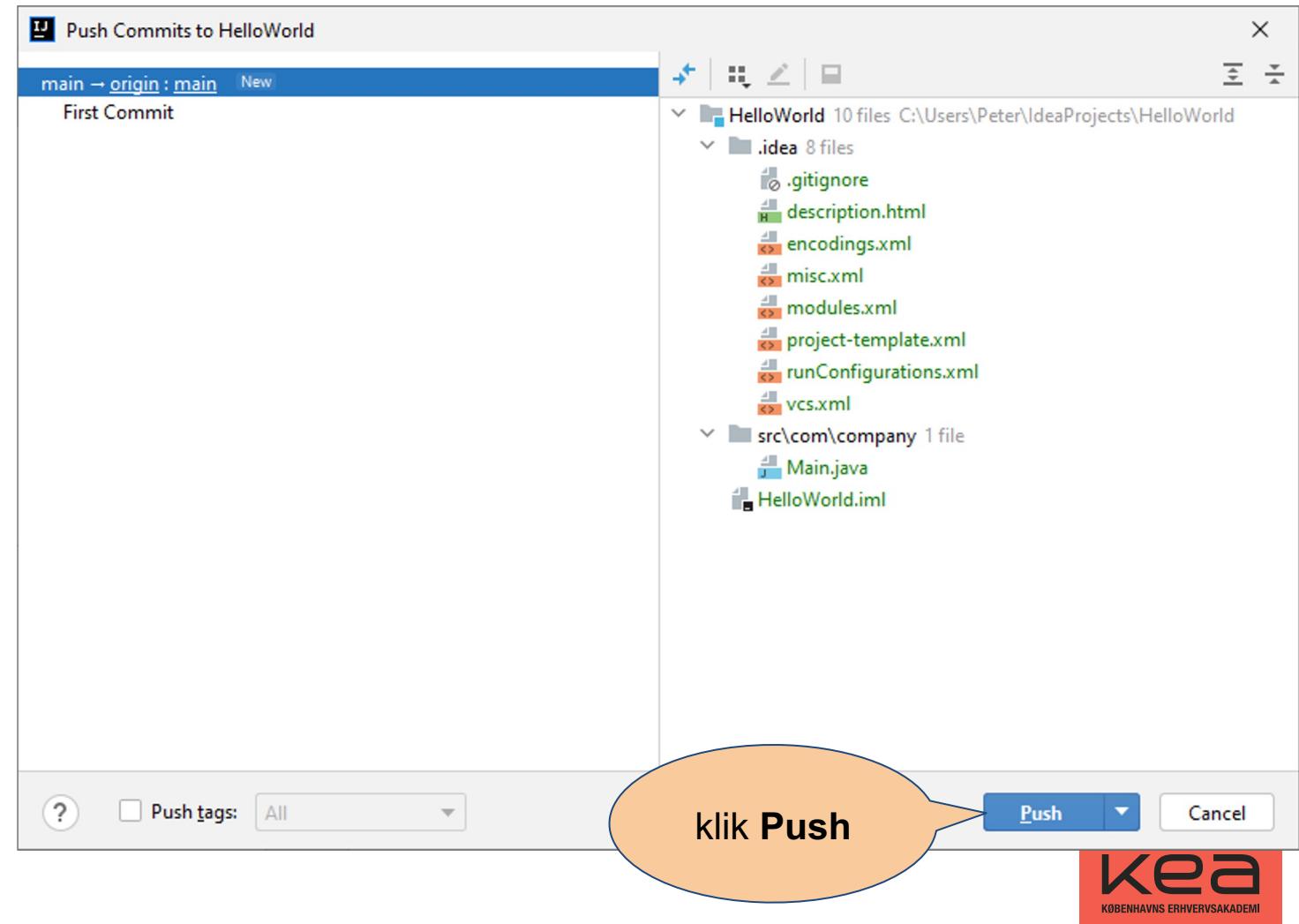


Den side kan du roligt lukke (men luk kun den ene tab - du har stadig brug for din GitHub side!)

SÅ KAN DU IGEN SE DIT PUSH-VINDUE

Her kan du se alle de filer du tilføjede til din commit - og ude til venstre den besked du skrev i forbindelse med committen.

Tryk på den blå Push-knap for at lægge det op på GitHub.



SÅ ER DER PUSHET

Nede i højre hjørne af IntelliJ kan du se en besked om at der er blevet pushet til "origin".

Det betyder at din kode nu ligger på GitHub, så skift over i browseren for at se det.

The screenshot shows the IntelliJ IDEA interface with a Java file named Main.java open. The code contains a simple 'Hello World' program. In the bottom right corner of the IDE, there is a notification box that says "Pushed main to new branch origin/main". The status bar at the bottom shows the time as 1:13, encoding as CRLF, character set as UTF-8, and spaces as 4 spaces. The file name "main" is also visible in the status bar. The bottom right corner features the KED logo (KØBENHAVNS ERHVERVSAKADEMI).

```
World - Main.java
Main.java x
package com.company;
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}

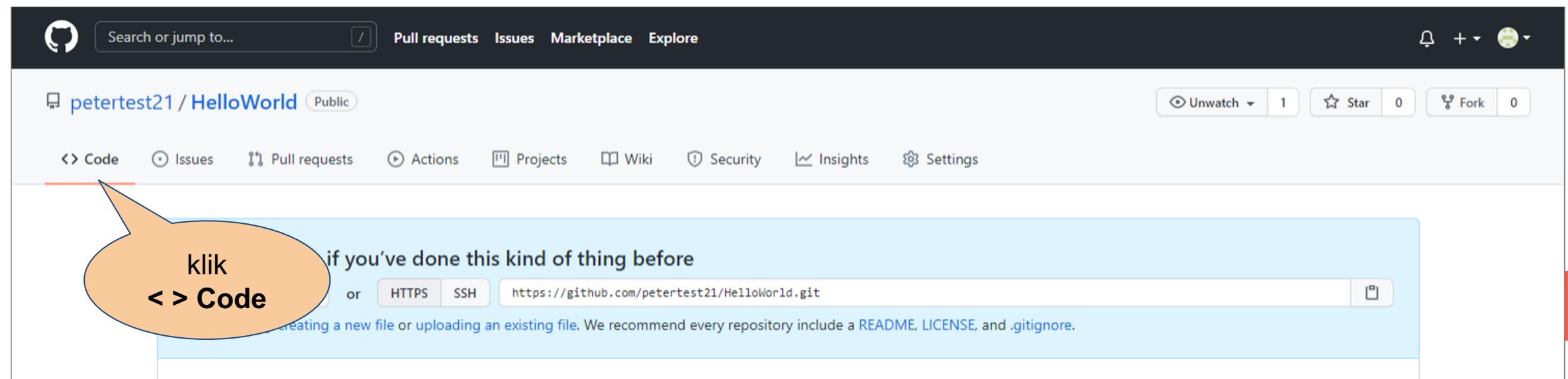
Pushed main to new branch origin/main
Event Log
1:13 CRLF UTF-8 4 spaces main
KED KØBENHAVNS ERHVERVSAKADEMI
```

```
package com.company;
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

ÅBN DIT GITHUB REPOSITORY I BROWSEREN

Hvis du er kommet til at lukke fanebladet, så har du stadig linket du kopierede til Remote. Og ellers kan du altid finde det, på github.com, når du er logget ind, kan du se dine repositories øverst til venstre.

Øverst på hver side under dit repository, er der en menu - vælg < > Code



HER KAN DU SE DIN KODE

Der er allerede kommet en del filer og mapper - og alt efter hvad du kaldte din package, så er der også en mappe med dens navn.

The screenshot shows a GitHub repository page. At the top, there's a dark header with the GitHub logo, a search bar, and navigation links for Pull requests, Issues, Marketplace, and Explore. Below the header, the repository name 'petertest21/HelloWorld' is shown, with a 'Public' badge. A navigation bar below the repository name includes links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. The 'Code' link is underlined, indicating it's the active tab. On the left, there's a sidebar with a 'main' branch dropdown showing '1 branch' and '0 tags'. To the right of the sidebar, there's a commit history table. The first commit is by 'petlatkea' and is titled 'First Commit'. It was made 5 minutes ago and includes three files: '.idea', 'src/com/company', and 'HelloWorld.iml', all of which are 'First Commit' and were made 5 minutes ago. Below the commit history, there's a note: 'Help people interested in this repository understand your project by adding a README.' followed by a green 'Add a README' button. An orange speech bubble with the text 'klik på src/ mappen' points to the 'src/com/company' file in the commit history table.

File	Commit	Time
.idea	First Commit	5 minutes ago
src/com/company	First Commit	5 minutes ago
HelloWorld.iml	First Commit	5 minutes ago

I SRC/PACKAGE/ MAPPEN LIGGER EN ENKELT FIL

A screenshot of a GitHub repository page for 'petertest21/HelloWorld'. The repository is public. The main tab is selected. A commit from 'petlatkea' titled 'First Commit' is shown, containing a single file named 'Main.java'. A blue speech bubble with the text 'klik på Main.java' points to this file.

A screenshot of the GitHub commit page for 'Main.java'. The commit is titled 'First Commit' by 'petlatkea'. The code content is displayed:

```
1 package com.company;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         System.out.println("Hello World");
7     }
8 }
```

A screenshot of the GitHub commit page for 'Main.java'. The commit is titled 'First Commit' by 'petlatkea'. The code content is displayed:

```
1 package com.company;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         System.out.println("Hello World");
7     }
8 }
```

TILLYKKE!

Du har nu installeret Git - oprettet en konto på GitHub - sat IntelliJ op til at bruge Git - givet IntelliJ rettigheder til at tilgå din konto på GitHub - og fået afprøvet at det hele virker!

Nu er det hele sat op - så fremover bliver det super-simpelt når du skal bruge Git.

Glæd dig!

A close-up photograph of a fluffy, light brown dog, possibly a Golden Retriever or Poodle mix, sitting on a dark-colored couch. The dog's head is bowed down, and it appears to be resting or sleeping. The background is slightly blurred.

PHEW!!

RET ERROR 403

Hvis du får en Error 403 i forsøget på at Authorize IntelliJ på Github, så følg de følgende trin, før du prøver igen.

Hvis du ikke har fået en Error 403, så er du kommet ind i en del af guiden som du ikke skal følge - gå tilbage igen.



Error 403: Not Authorized

This request is only valid within same origin
Try to [sign in as another user](#)?



petlatkea

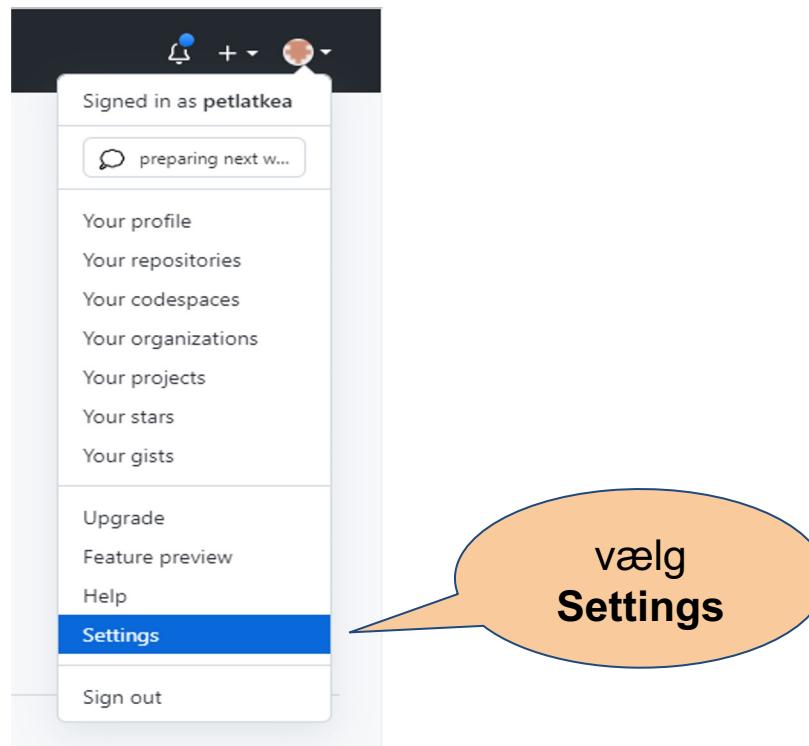
Your personal account

[Switch to another](#)

RET ERROR 403 – FJERN INTELLIJ FRA GITHUB

Gå ind i Settings på din GitHub konto

og vælg derefter "Applications",
et af de nederste menu-punkter på
settings-siden



og derefter
Applications

[Public profile](#)

[Account](#)

[Appearance](#)

[Accessibility](#)

[Notifications](#)

Access

[Billing and plans](#)

[Emails](#)

[Password and authentication](#)

[SSH and GPG keys](#)

[Organizations](#)

[Moderation](#)

Code, planning, and automation

[Repositories](#)

[Packages](#)

[GitHub Copilot](#)

[Pages](#)

[Saved replies](#)

Security

[Code security and analysis](#)

Integrations

[Applications](#)

[Scheduled reminders](#)

Archives

[Security log](#)

RET ERROR 403 – FJERN INTELLIJ FRA GITHUB

Under Applications - vælg fanebladet "Authorized OAuth Apps"

Der kan du se JetBrains IDE Integration - tryk de tre prikker og vælg Revoke.

Applications

The screenshot shows the GitHub Applications page. At the top, there are three tabs: 'Installed GitHub Apps', 'Authorized GitHub Apps', and 'Authorized OAuth Apps'. The 'Authorized OAuth Apps' tab is highlighted with a light blue background. Below the tabs, a message says 'You have granted 1 application access to your account.' To the right are 'Sort' and 'Revoke all' buttons. The main list contains one item: 'JetBrains IDE Integration' with a small icon. Below the app name, it says 'Last used within the last week · Owned by JetBrains'. To the right of the app name is a vertical ellipsis menu with options 'Report abuse' and 'Revoke'. A large orange callout bubble points to the 'Revoke' button with the text 'Tryk Revoke - og bekræft når du bliver bedt om det.'

You have granted 1 application access to your account.

Sort ▾ [Revoke all](#)

[JetBrains IDE Integration](#)
Last used within the last week · Owned by JetBrains

Report abuse

⋮

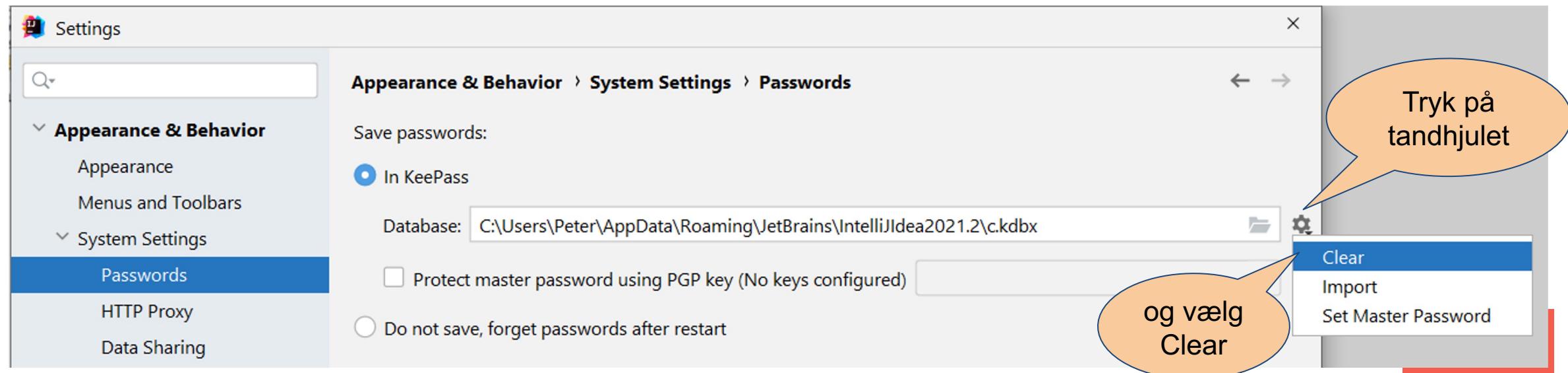
Revoke

ⓘ Read more about connecting with third-party applications at [GitHub Help](#).

RET ERROR 403 – FJERN PASSWORDS FRA INTELLIJ

Når du har fjernet JetBrains IDE Integration fra GitHub, så åbn IntelliJ og gå ind i Settings / Preferences.

Vælg Appearance & Behavior > System Settings > Passwords, og tryk på det lille tandhjul ved siden af Database-feltet, og vælg Clear



RET ERROR 403 – TJEK DIN GIT EMAIL

Åbn terminalen på Mac eller Git-Bash på Windows, og skriv:

```
git config --global user.email
```

tryk enter, og verificer at den udskriver den samme email-adresse som du registrerede dig med på GitHub.

Hvis ikke, så skriv:

```
git config --global user.email "den_rigtige@email.adresse"
```

og tryk enter.

husk "
tegn

skriv den
email du
registrerede
dig med

RET ERROR 403 – PRØV IGEN

Så er du klar til at prøve igen.

Hvis IntelliJ står og venter, så cancel, og vælg Push fra Git-menuen igen.

Fortsæt så med [Log In via GitHub...](#) og se om det virker bedre anden gang.

Hvis ikke, så bed om hjælp!