

GitHub for Windows och GitShell

En introduktion till programmen och de första grunderna i git.

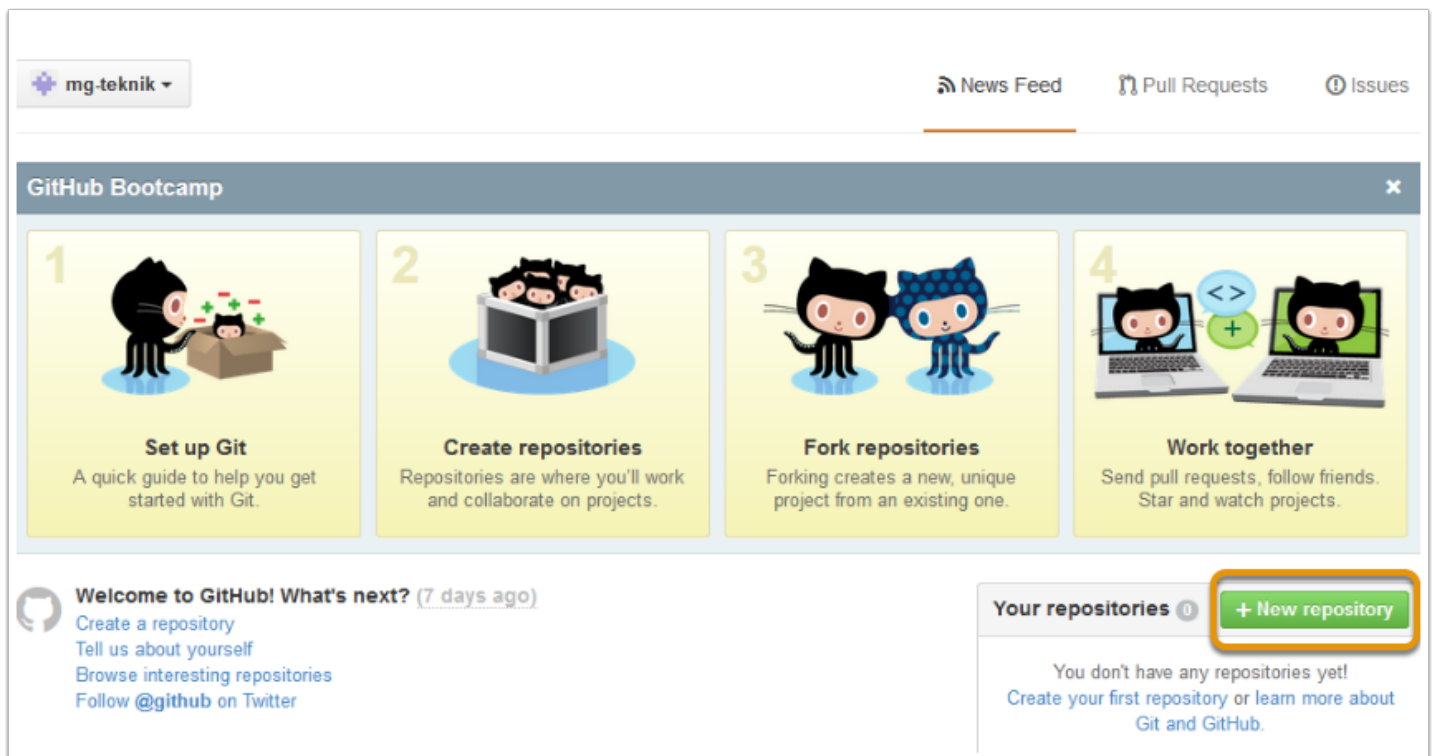
Det finns en tidigare introduktion. Den visar hur man skapar konto på git och använder GitHub for Windows för att synka ett repo.

Här upprepar vi vissa delar, men går även igenom de viktigaste kommandona för att jobba med git i GitShell, versionen som GitHub for Windows installerar.

Skapa repo på GitHub


Behöver du hjälp att skapa konto på GitHub, kolla en tidigare instruktion.

Vi skapar ett första repo. T.ex. genom att klicka på den markerade knappen.



Ge namn åt repot och skapa readme


Välj ett namn för ditt repo. Kryssa även i alternativet för att skapa en README, det är god standard.


Owner  **mg-teknik** ▾

Repository name ✓

Great repository names are short and memorable. Need inspiration? How about [this](#)

Description (optional)

☒  **Public**
Anyone can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

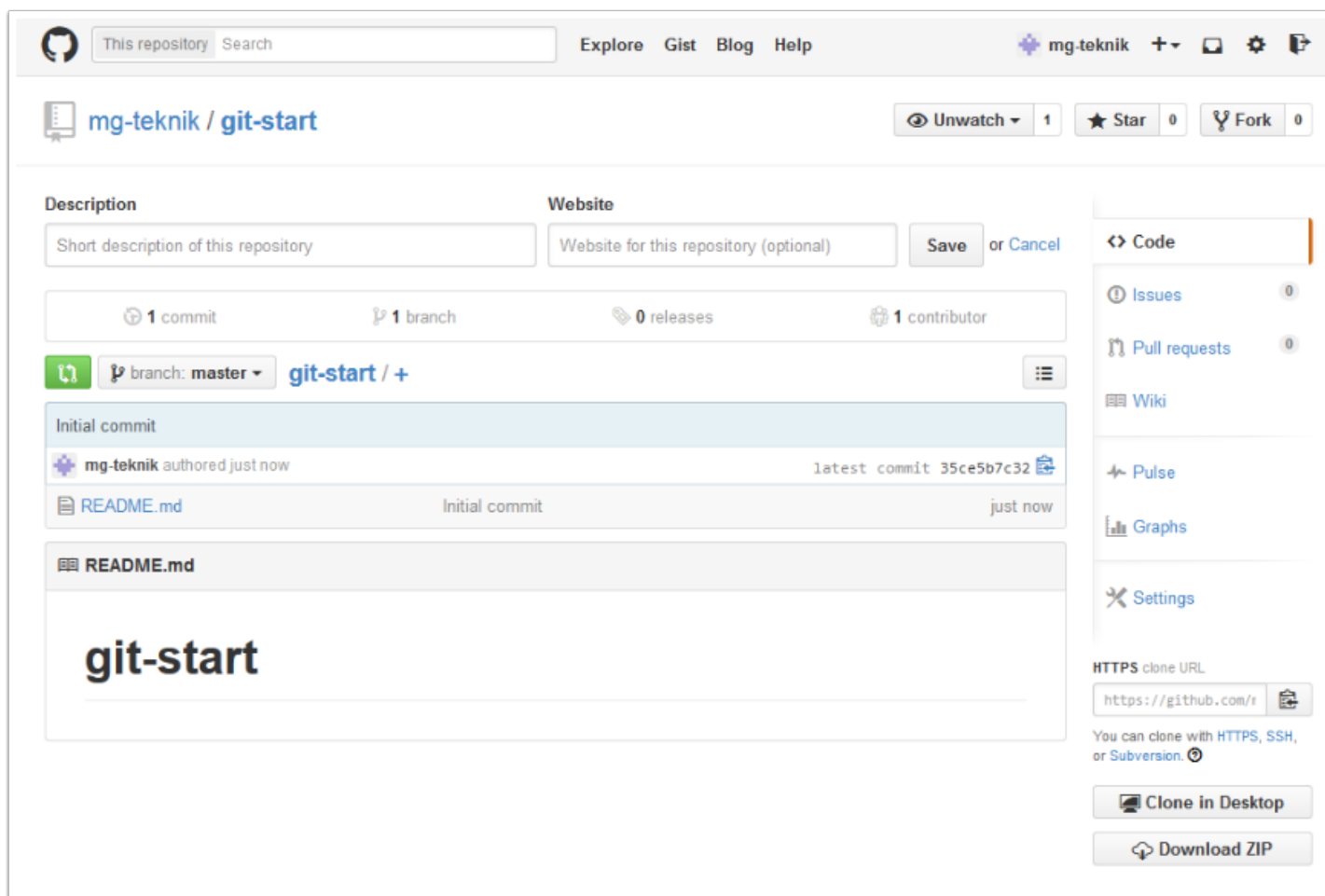
☒ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer. Skip this step if you already have a README.

Add .gitignore: **None** ▾ | Add a license: **None** ▾ ⓘ

Create repository

Så här ser första repot ut

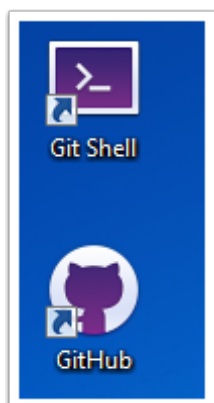
Ditt första repo på GitHub ser ut så här.



Två installerade program

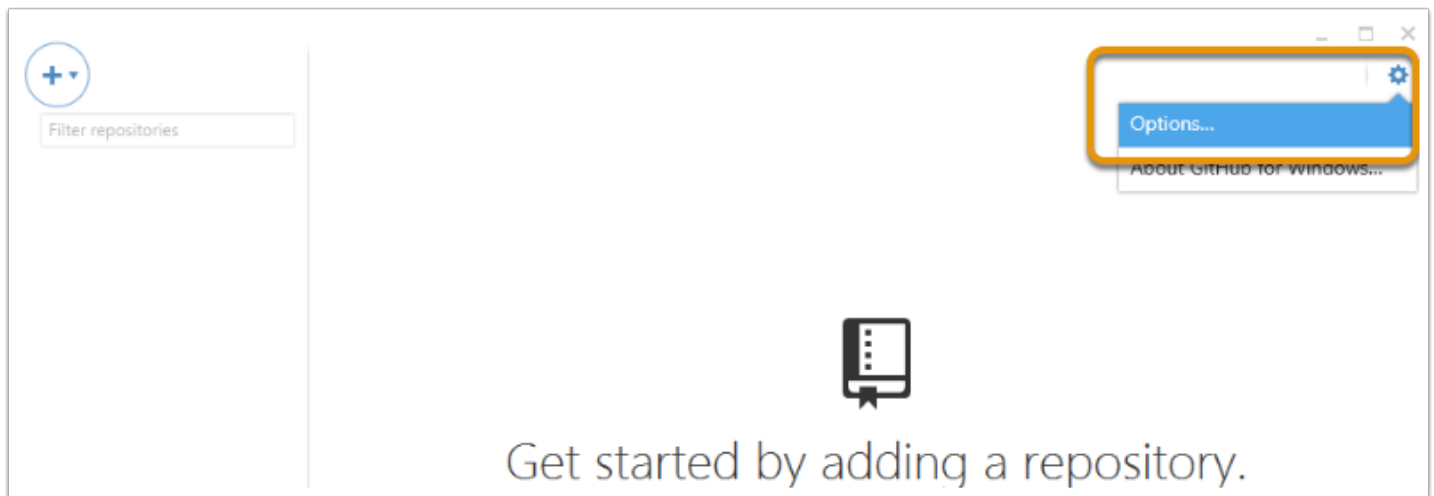
När vi installerade GitHub for Windows fick vi även med programmet Git Shell.

Nu startar vi GitHub.



Gränssnittet för GitHub for Windows

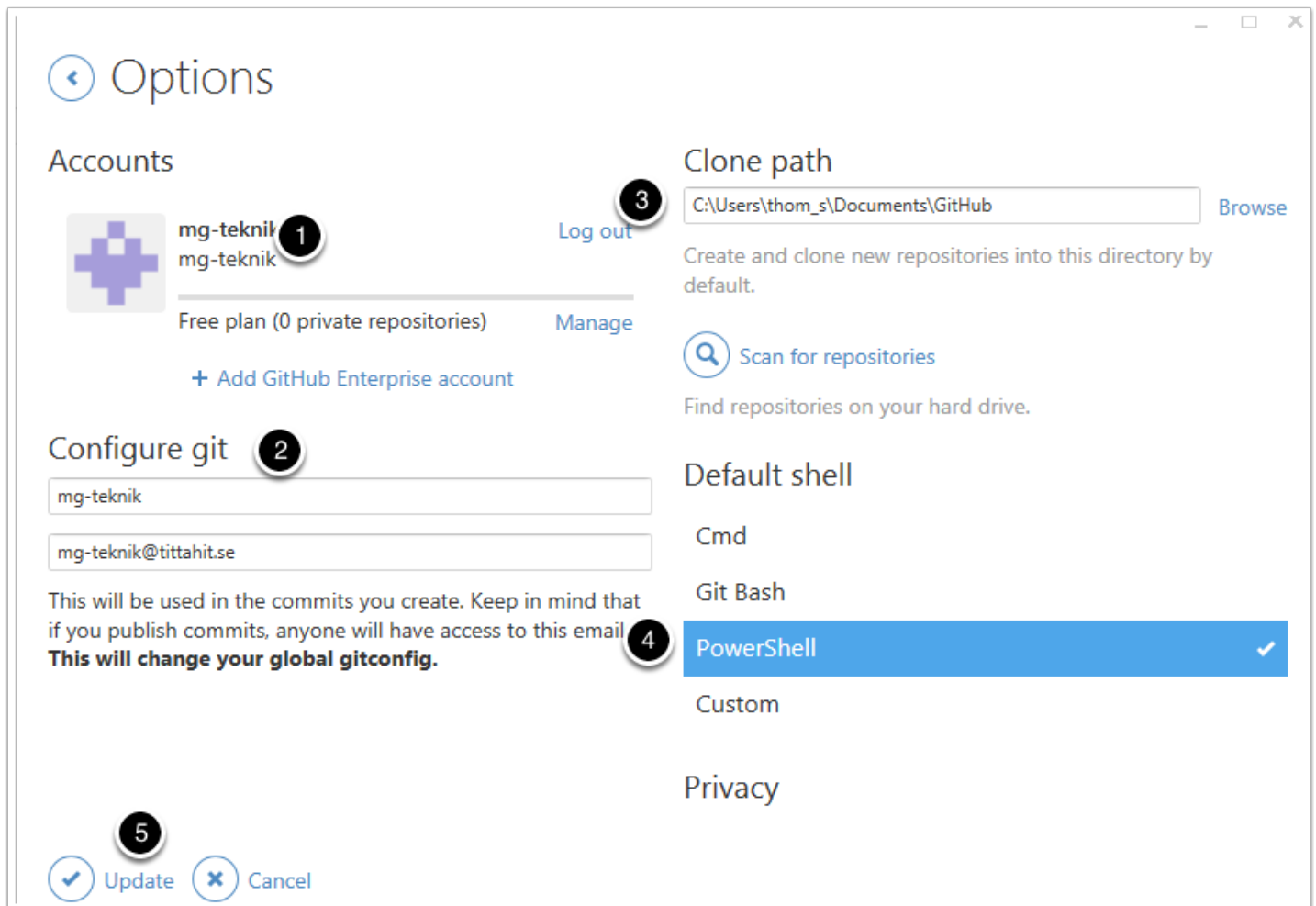
Programmet är rent när vi startar det. Klicka på kugghjulet och välj Options.



Alternativ för GitHub for Windows

Här ser vi en del inställningar - som går att ändra:

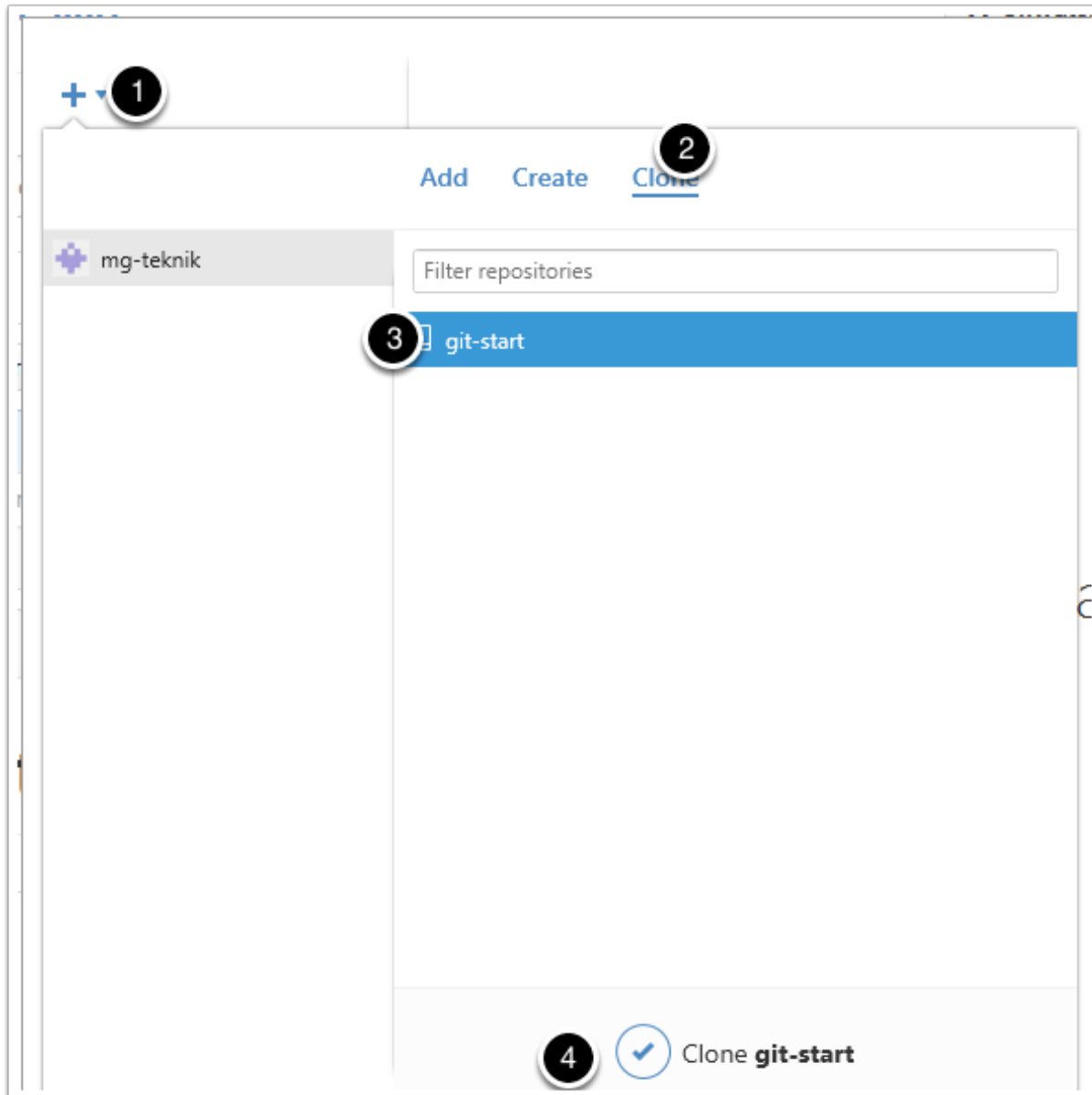
1. Vilket konto som är kopplat till programmet
2. Vilken e-post som är kopplad
3. Vilken mapp på datorn vi har de lokala filerna i
4. Vilket Shell-program som är förvalt.
5. Klicka här för att avsluta eller uppdatera ändringar.



Klona ett repo från GitHub

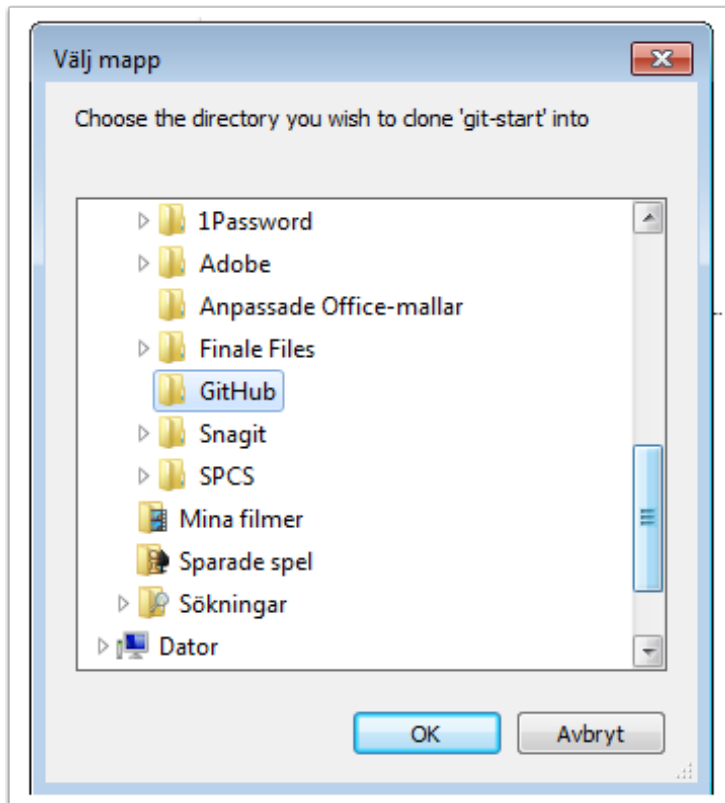
Klicka på +, välj repo och sen Clone.

Vi har enbart ett repo så vi väljer att klona det. Det innebär att vi kopierar originalet, som alltid finns på GitHub.



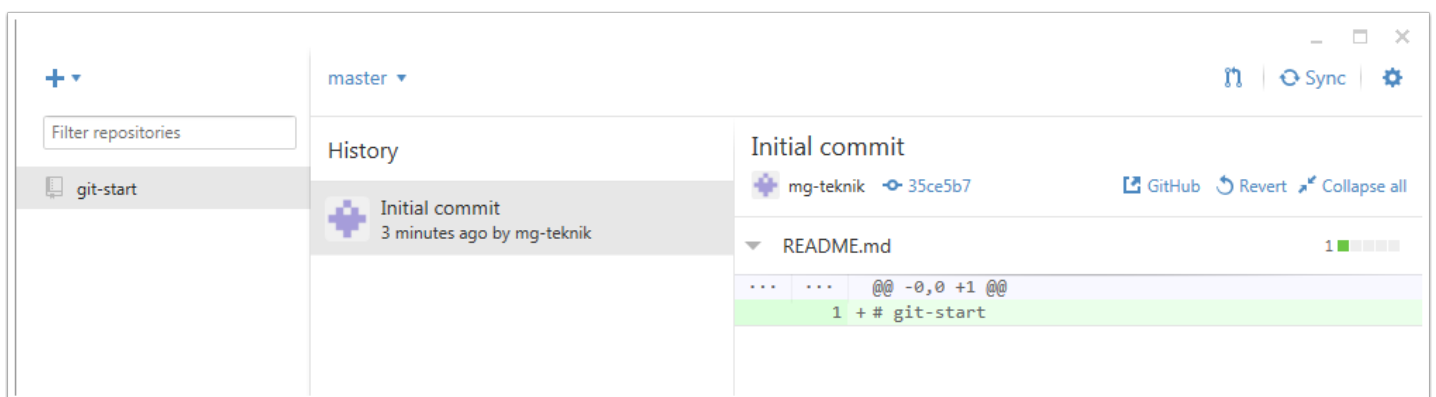
Placera repot lokalt

Här får vi frågan om var det ska placeras lokalt på datorn. Vi väljer den mapp GitHub föreslagit.



Repot i GitHub

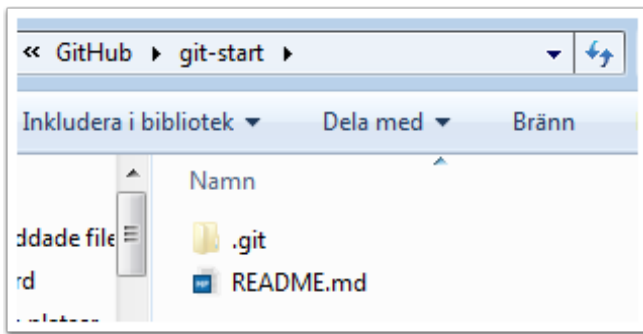
Så här ser repot ut i GitHub for Windows



Öppna README.md för redigering

Gå till mappen och öppna filen README.md för redigering.

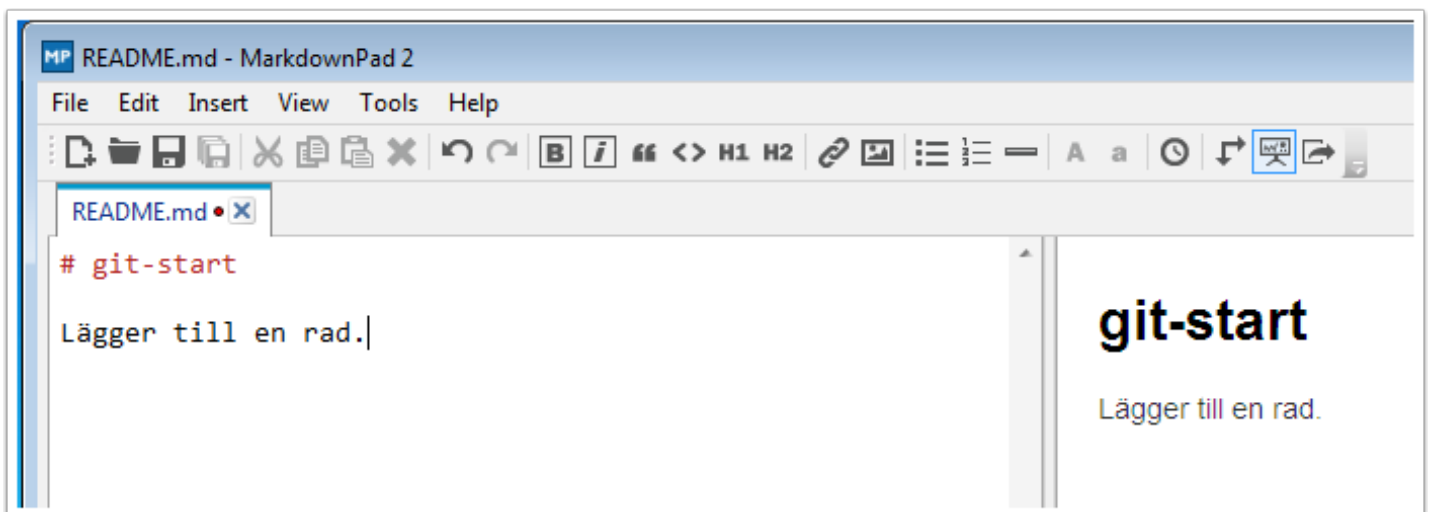
Filändelsen .md står för markdown och kan öppnas med vanlig textredigeringsprogram, t.ex. [Notepad++](#). Vill man ha ett gratis program för just markdownfiler finns t.ex. [MarkdownPad](#).



Ändra i dokumentet

Vi lägger till en rad med programmet MarkdownPad (som är förvalt för markdown-filer på denna datorn).

Spara filen.



Ändringar i GitHub for Windows

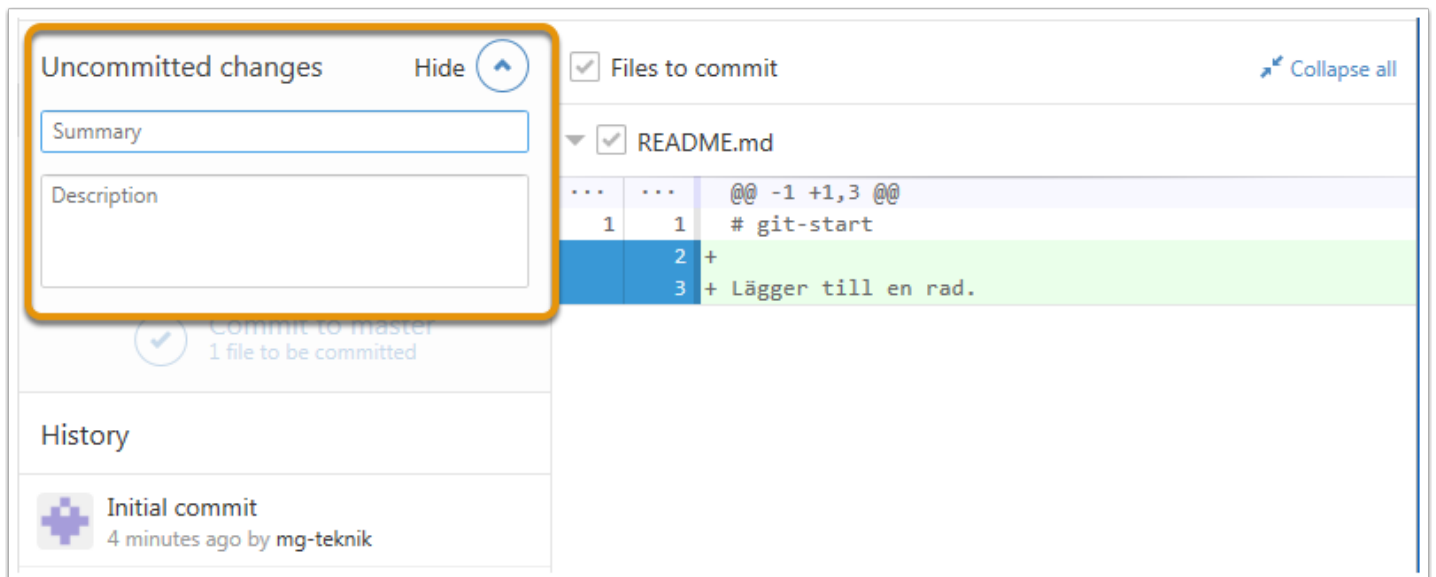
I GitHub for Windows finns en rad som säger "Uncommitted changes". Det har skett ändringar i repot.

Klicka på knappen "Show" för att se dem och lägga till.



Ändringarna syns

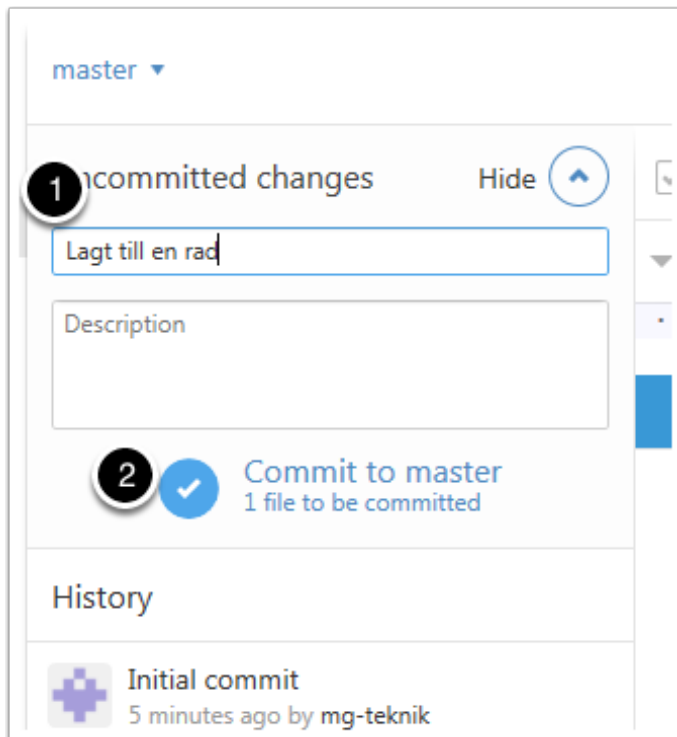
Vi ser ändringarna till höger. Men vi ser också att de är "uncommitted" - inte tillagda. GitHub for Windows har skapat en ruta som vi bara behöver fylla i.



Beskriv ändringarna

Vi beskriver ändringarna. Det är bra att vara tydlig, den gång man vill komma tillbaka till just detta stadium av projektet.

Klicka på "Commit"



Tillagda - men inte synkade ändringar

Nu ser vi att rubriken ändrats till "Unsynced changes". Ändringarna är tillagda i det lokala repot - men de är inte synkade med repot på GitHub. Klicka på "Sync"



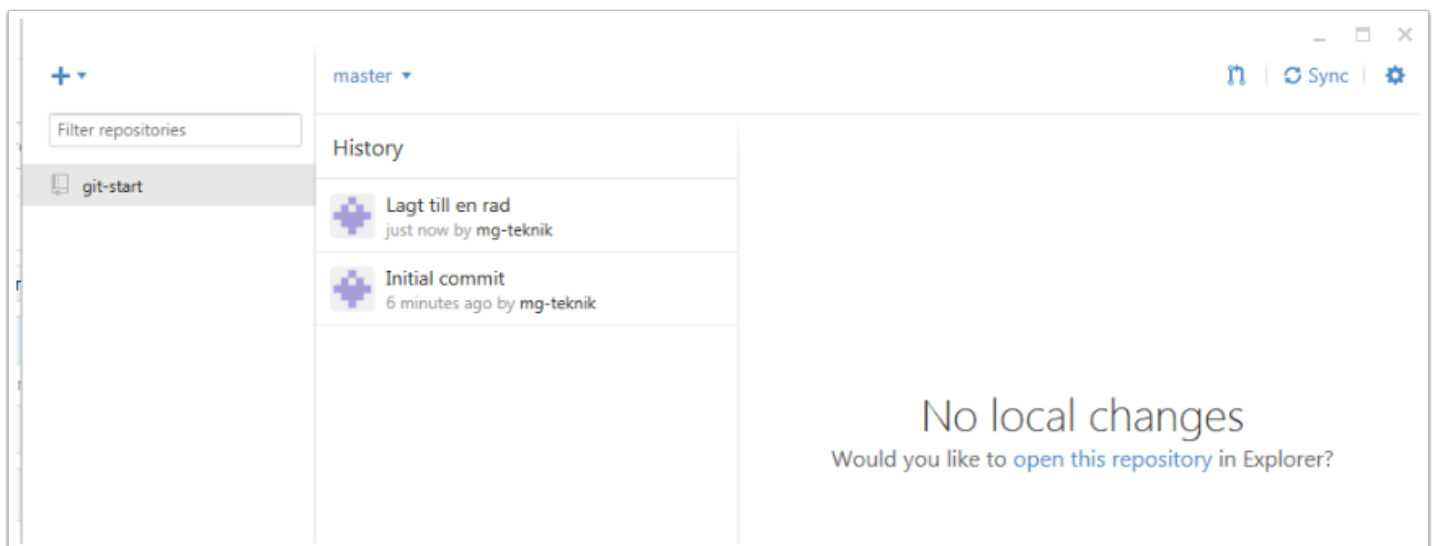
Synkning sker

Den blå linjen visar hur synkningen fortskrider. Här går det snabbt.



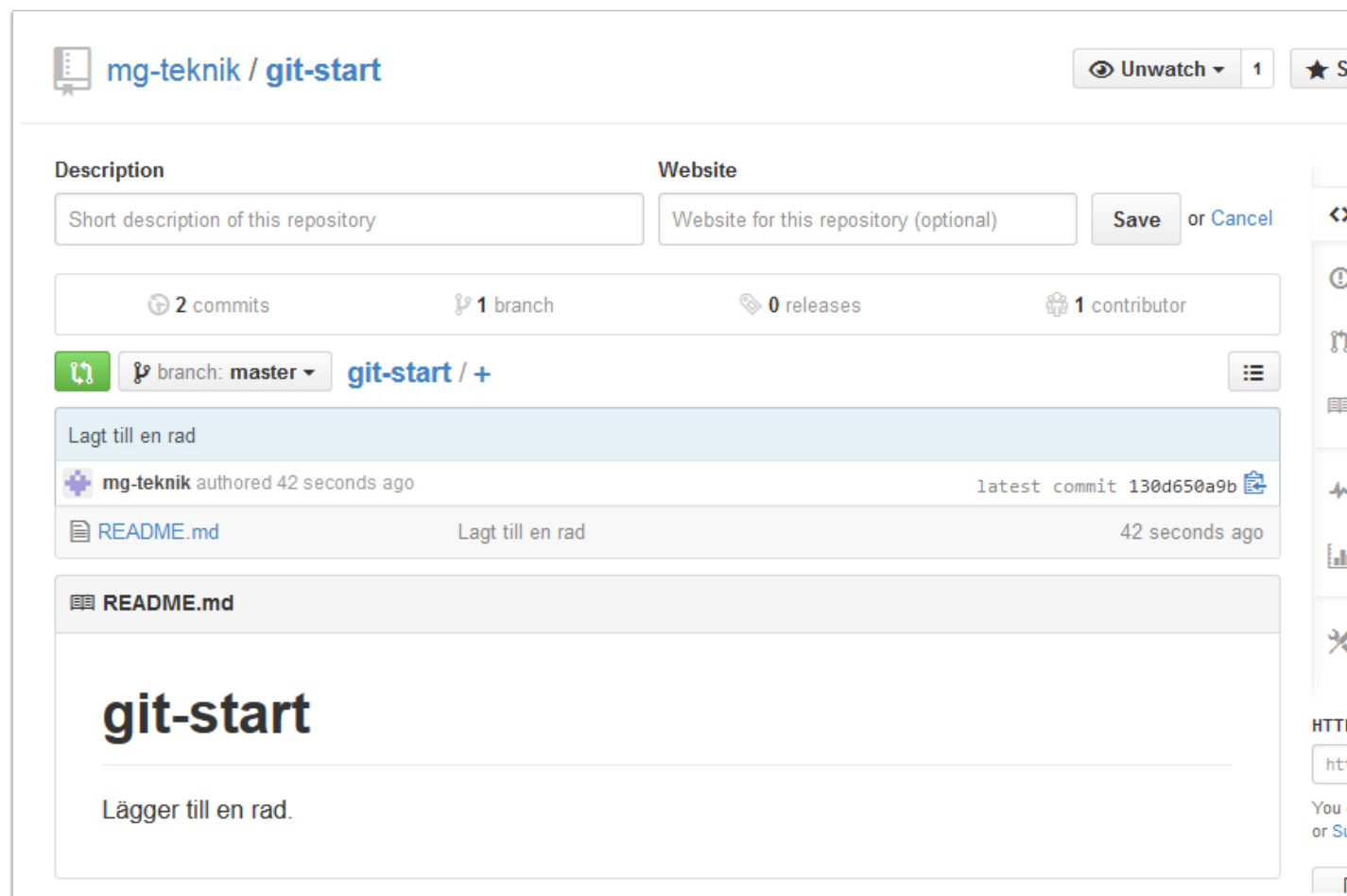
Synkat och klart

När allt är synkat ser det ut så här. Inget som pöckar på uppmärksamhet.



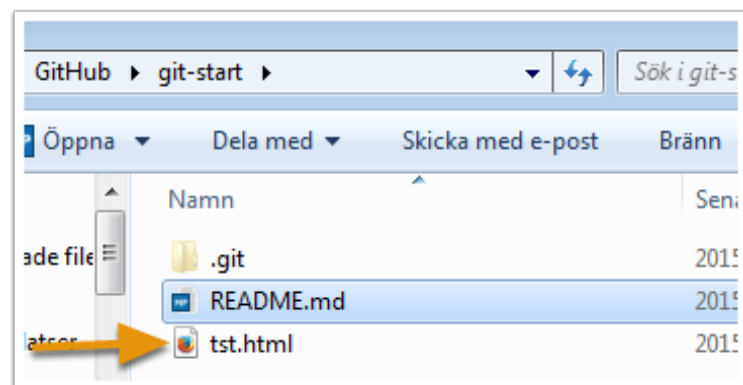
Kontrollera på GitHub

När vi går till GitHub ser vi att raden vi lade till finns med.



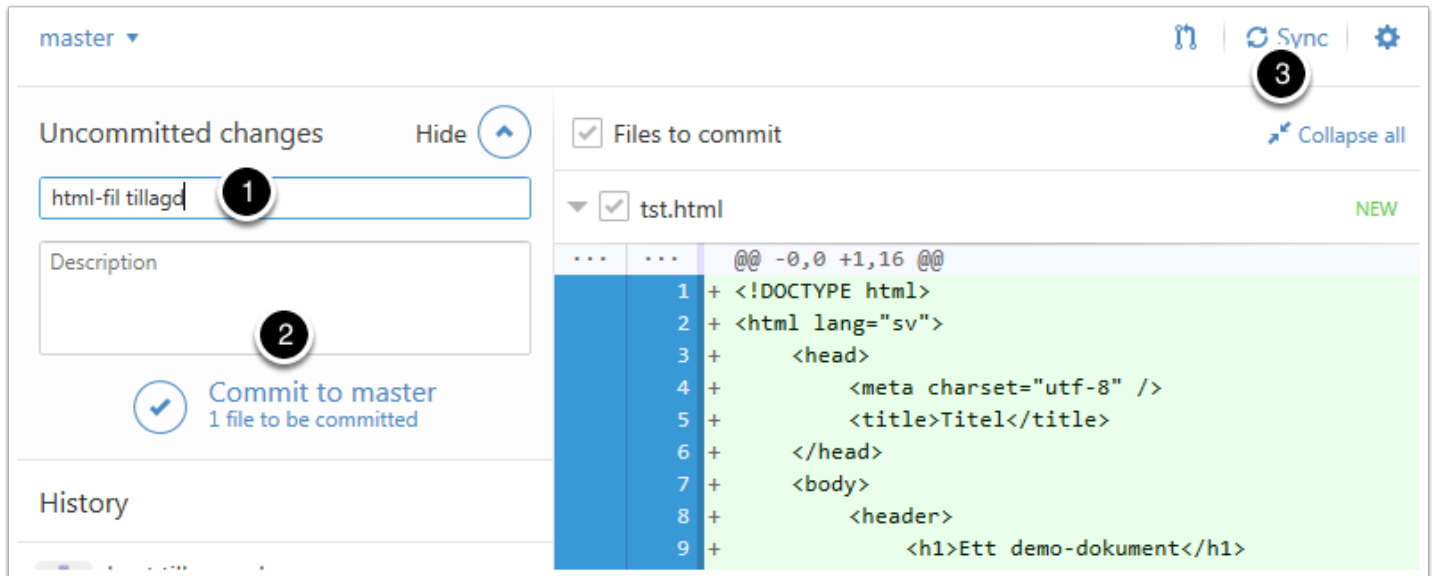
Lägg till ytterligare en fil

Vi lägger till en fil, tst.html




Lägg till lokala repot via GitHub for Windows

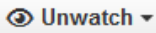
Vi ger ändringen ett namn och "comittar" och synkar.



Kontrollerar på GitHub

Vi kollar att filen finns på GitHub.

 **mg-teknik / git-start**

 Unwatch ▾ **1**

Description


Short description of this repository


Website


Website for this repository (optional)


Save



 or [Cancel](#)


 **3** commits

 **1** branch


 **0** releases


 **1** contributor


  branch: **master** ▾ **git-start / +**



html-fil tillagd


 **mg-teknik** authored 14 seconds ago

latest commit **53d49ef7c6** 

 [README.md](#)


Lagt till en rad

3 minutes ago

 [tst.html](#)

html-fil tillagd

14 seconds ago

 **README.md**

git-start

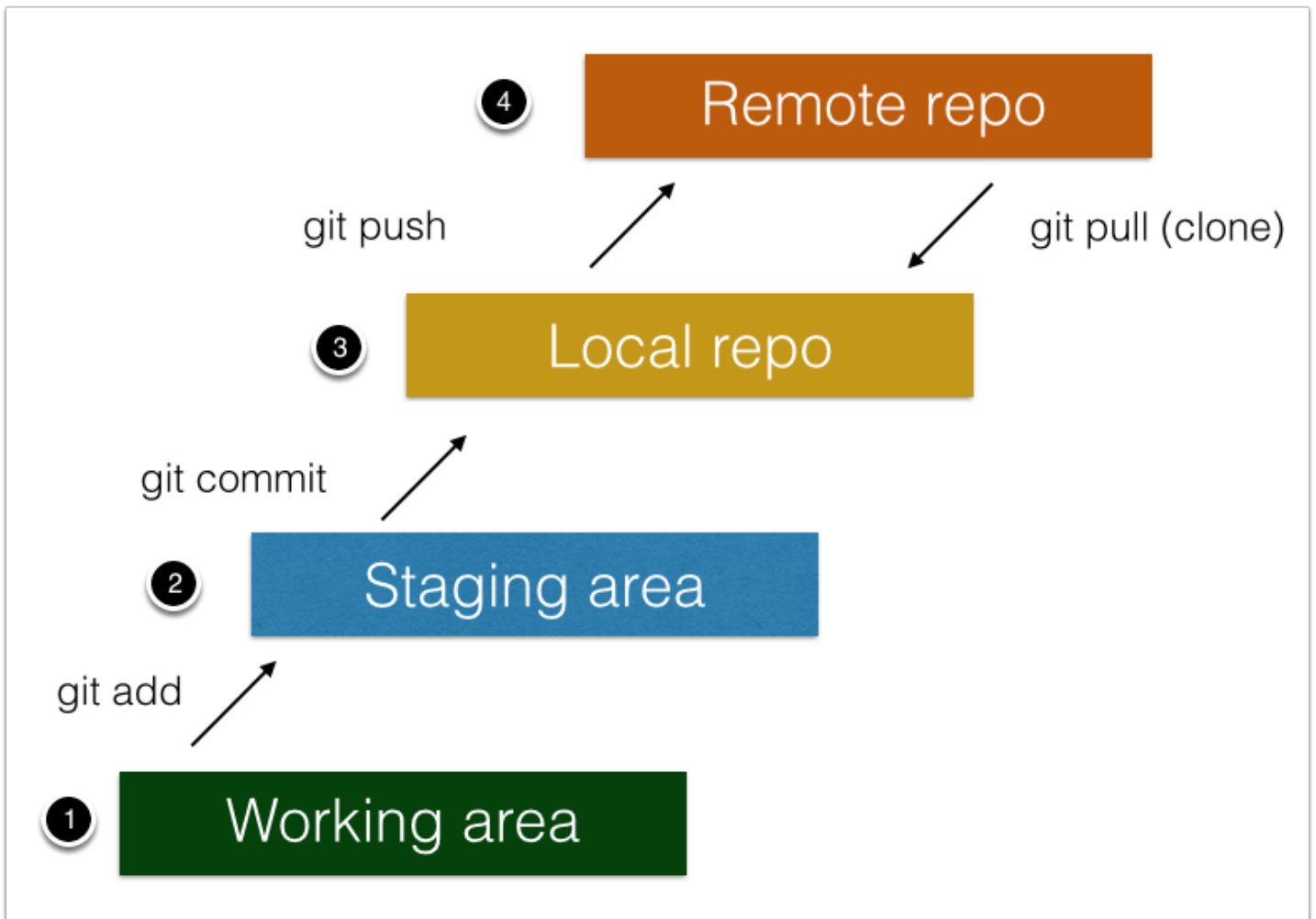
Lägger till en rad.

Översikt över gits nivåer och några kommandon

Här ser vi de olika nivåerna för arbete med git.

1. Working area - de lokala filerna vi jobbar med i den valda mappen
2. Staging area - filer vi jobbat med och markerat klara att lägga till git (`git add`)
3. Local repo - filer vi lagt till gits databas (`git commit`)
4. Remote repo - originalet dit vi lägger våra ändringar (och där vi hämtar original)

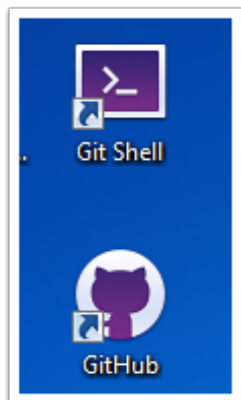
GitHub for Windows gör `add` och `commit` i ett svep, men med Git Shell behöver man göra båda var för sig.



Git Shell

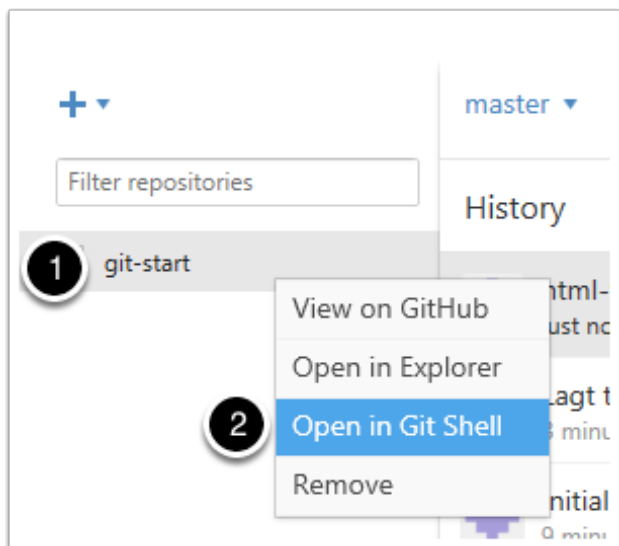
GitHub for Windows installerade två program. Nu är det dags att prova Git Shell. Med det lär vi oss de vanligaste kommandona i git mer handgripligt, vi måste nämligen skriva dem.

Vi kan öppna det från skrivbordet. Men det är betydligt enklare att öppna inifrån GitHub for Windows.



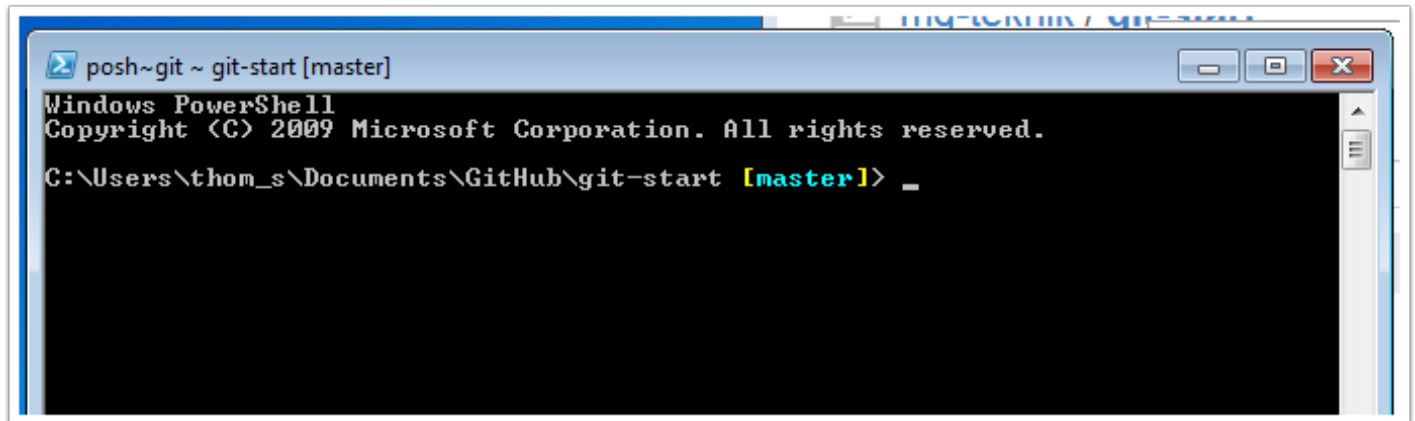
Öppna Git Shell

Högerklicka på repot och välj "Open in Git Shell". Finessen är att vi då hamnar direkt i repot. Det kan vara lite krångligt att ta sig dit i kommandotolken annars. Man behöver kunna fler kommando i så fall.



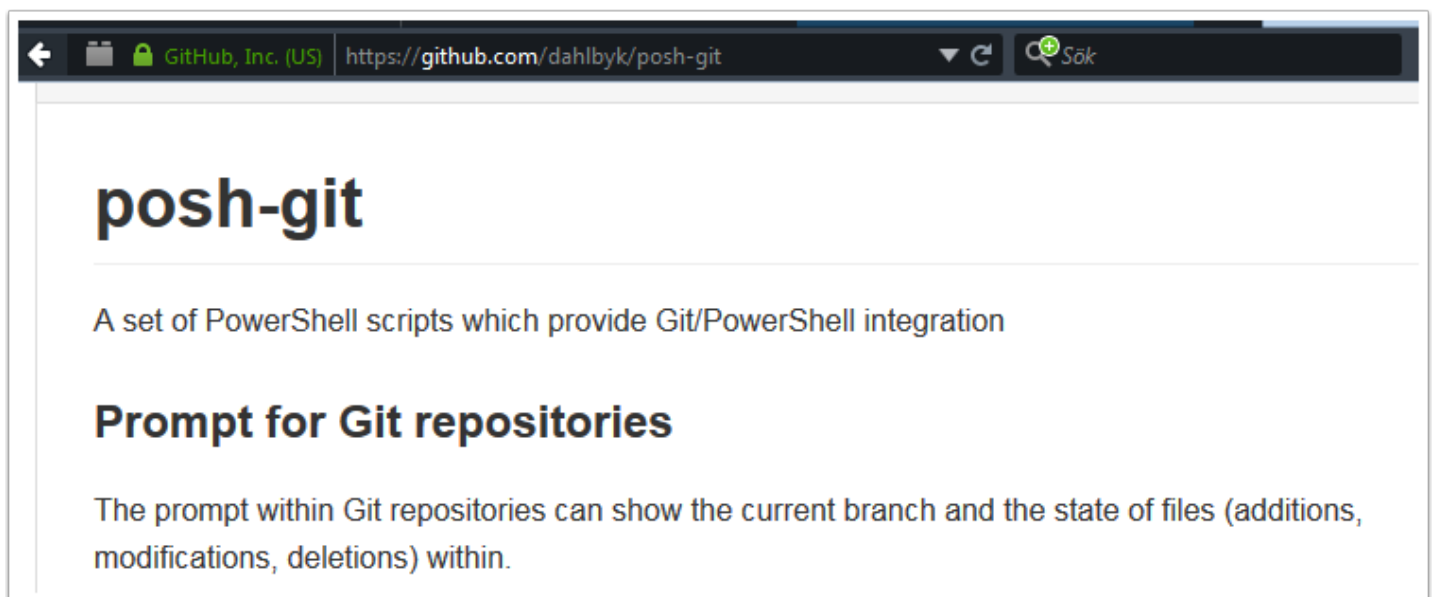
Posh-git

Git Shell finns i en version där man använder ett repo från git som heter Posh-git. Det ser ut så här när vi startar.



Posh-git

Det är i sin tur ett repo på GitHub -



Annorlunda prompt

Finessen med posh-git är att den har en liten annorlunda prompt ("raden" som syns framför platsen vi skriver in kommandon). Där finns inbyggd hjälp att få översikt över hur det står till i repot man befinner sig, det vi ser inom hakparenteserna. Nästa bild visar lite vad prompten talar om, men vi fördjupar oss inte i det här.

Git Shell är för mer avancerade användare, men vi går igenom grunderna, för de ger .

(GitHub for Windows är snäll och gör det enklare för oss. Där kan vi t.o.m. hoppa över en del steg. Så för helhetens skull är det viktigt att prova på Git Shell, eftersom det ger en pedagogisk, handfast första inblick i hur git fungerar.)

The Prompt

PowerShell generates its prompt by executing a `prompt` function, if one exists. posh-git defines such a function in `profile.example.ps1` that outputs the current working directory followed by an abbreviated `git status`:

```
C:\Users\Keith [master]>
```

By default, the status summary has the following format:

```
[{HEAD-name} +A ~B -C !D | +E ~F -G !H !]
```

- `{HEAD-name}` is the current branch, or the SHA of a detached HEAD
 - Cyan means the branch matches its remote
 - Green means the branch is ahead of its remote (green light to push)
 - Red means the branch is behind its remote
 - Yellow means the branch is both ahead of and behind its remote
- ABCD represent the index; EFGH represent the working directory
 - `+` = Added files
 - `~` = Modified files
 - `-` = Removed files
 - `!` = Conflicted files
 - As in `git status`, index status is dark green and working directory status is dark red
 - The trailing `!` means there are untracked files

git status

Det kanske vanligaste kommandot man använder är *git status*.

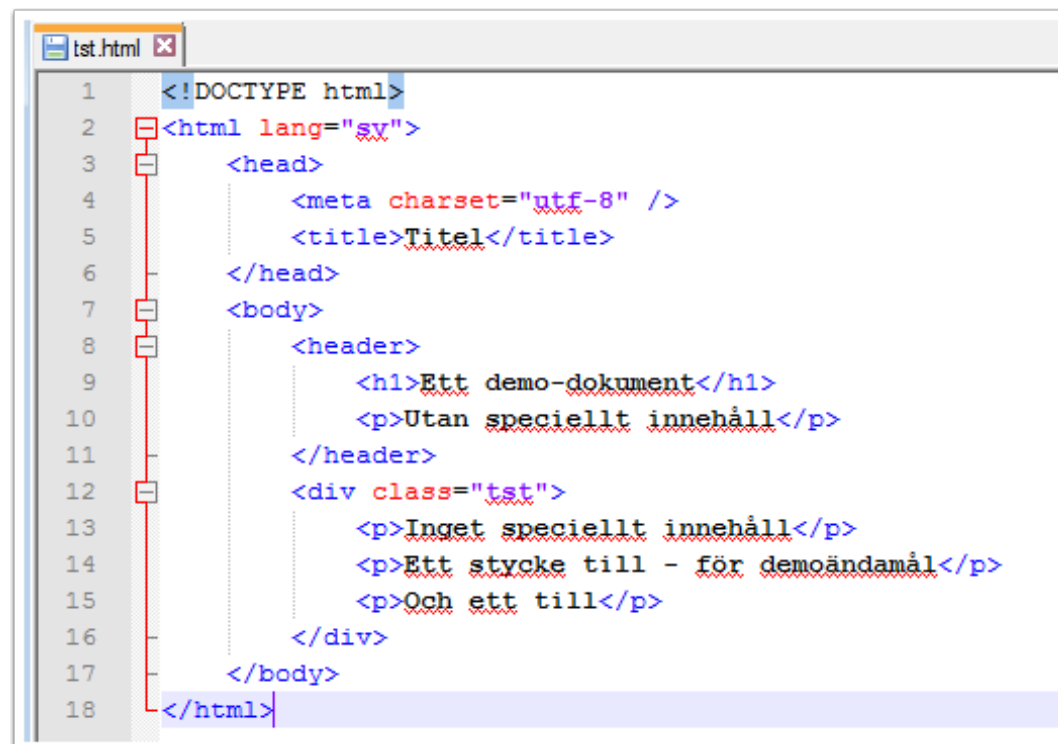
Alla git-kommandon inleds med git följt av mellanslag. Status talar om hur det lokala repot ser ut på lite olika sätt.

```
C:\Users\thom_s\Documents\GitHub\git-start [master]> git status
On branch master
Your branch is up-to-date with 'origin/master'.

nothing to commit, working directory clean
C:\Users\thom_s\Documents\GitHub\git-start [master]>
```

html-filen före redigering

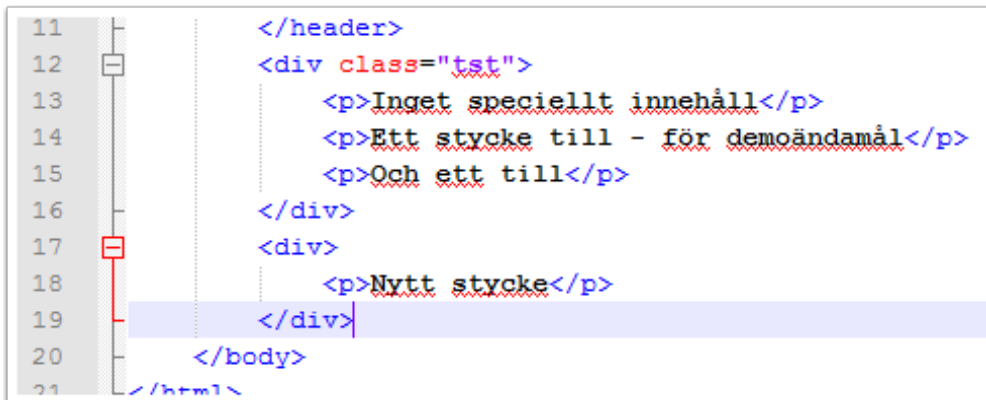
Så här ser filen vi lagt till ut.



```
1 <!DOCTYPE html>
2 <html lang="sv">
3   <head>
4     <meta charset="utf-8" />
5     <title>Titel</title>
6   </head>
7   <body>
8     <header>
9       <h1>Ett demo-dokument</h1>
10      <p>Utan speciellt innehåll</p>
11    </header>
12    <div class="tst">
13      <p>Inget speciellt innehåll</p>
14      <p>Ett stycke till - för demoändamål</p>
15      <p>Och ett till</p>
16    </div>
17  </body>
18 </html>
```

html-filen efter redigering

Vi lägger till ett par rader (17-19), en div-tag.



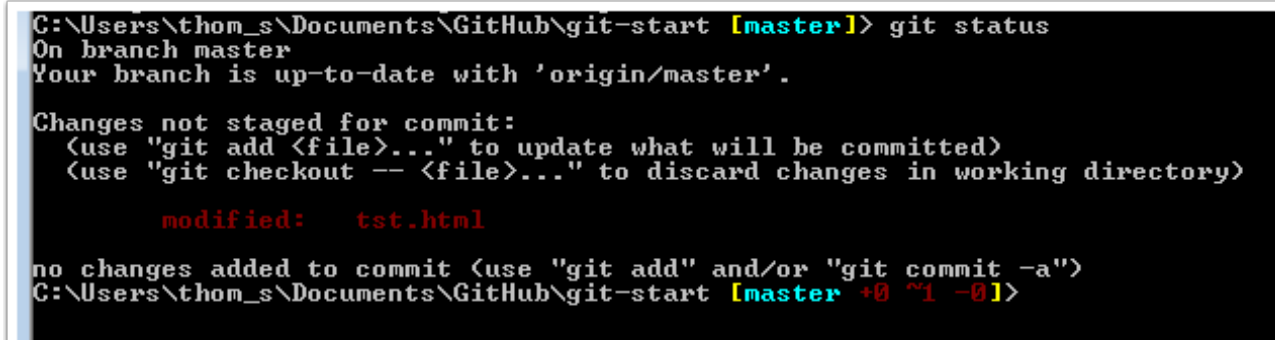
```
11 </header>
12 <div class="tst">
13   <p>Inget speciellt innehåll</p>
14   <p>Ett stycke till - för demoändamål</p>
15   <p>Och ett till</p>
16 </div>
17 <div>
18   <p>Nytt stycke</p>
19 </div>
20 </body>
21 </html>
```

git status

Vi kontrollerar i Git Shell med kommandot *git status*. Vi får veta följande:

Det finns ändringar som inte är tillagda (staged) för att kunna comittas. Det markeras genom den röda texten. Det ser vi även inom hakparentesen i slutet på prompten.

Vi får dessutom veta vilket kommando vi ska använda för detta: *add*



```
C:\Users\thom_s\Documents\GitHub\git-start [master] > git status
On branch master
Your branch is up-to-date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   tst.html

no changes added to commit (use "git add" and/or "git commit -a")
C:\Users\thom_s\Documents\GitHub\git-start [master +0 ~1 -0] >
```

git add

Med kommandot *add* gör vi filerna redo för en commit, dvs. de kan läggas i det lokala repot.

Man kan skriva *git add tst.html* för att bara lägga till just denna fil. Men har man flera filer man vill lägga till kan man använda syntaxen:

git add . (punkten betyder: lägg till alla filer) och det gör vi. (Här kommer lite varningar också, men de bryr vi oss inte om.)

Sen kör vi *git status* igen. Då ser vi att filen *tst.html* nu är grön. Se även prompten. Det innebär, som texten säger, att den är klar för en commit.

```
C:\Users\thom_s\Documents\GitHub\git-start [master +0 ~1 -0]> git add .
warning: LF will be replaced by CRLF in tst.html.
The file will have its original line endings in your working directory.
C:\Users\thom_s\Documents\GitHub\git-start [master +0 ~1 -0]> git status
warning: LF will be replaced by CRLF in tst.html.
The file will have its original line endings in your working directory.
On branch master
Your branch is up-to-date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   tst.html

C:\Users\thom_s\Documents\GitHub\git-start [master +0 ~1 -0]>
```

git commit -m "meddelande"

Syntaxen för en commit är: *git commit -m "Text om ändring som gjorts"*

Här får vi återigen lite varningar. Vi hoppar över dem och konstaterar att ändring gjorts. Se den gröna "master" i prompten.

```
C:\Users\thom_s\Documents\GitHub\git-start [master +0 ~1 -0]> git commit -m "änd
rat i html-fil"
[master warning: LF will be replaced by CRLF in tst.html.
The file will have its original line endings in your working directory.
b753b091 ändrat i html-fil
warning: LF will be replaced by CRLF in tst.html.
The file will have its original line endings in your working directory.
 1 file changed, 3 insertions(+)]

Warning: Your console font probably doesn't support Unicode. If you experience s
trange characters in the output, consider switching to a TrueType font such as L
ucida Console!
C:\Users\thom_s\Documents\GitHub\git-start [master]>
```

Kontrollera i GitHub for Windows

Öppnar vi GitHub for Windows ser vi att ändringarna syns i klartext. Men vi ser också att de är "unsynced". De är inte synkade med GitHub .

The screenshot shows the GitHub for Windows application interface. At the top, the 'master' branch is selected. On the right, there are icons for 'Sync' (with a '+1' badge) and 'Settings'. The main area is divided into two panes. The left pane, titled 'Unsynced changes', shows a list of changes: 'ändrat i html-fil just now by mg-teknik', 'Lagt till html-fil igen 4 minutes ago by mg-teknik', 'tagit bort html-fil 5 minutes ago by mg-teknik', 'html-fil tillagd 13 minutes ago by mg-teknik', and 'Lagt till en rad 16 minutes ago by mg-teknik'. The right pane, titled 'ändrat i html-fil', shows a diff view for the file 'tst.html'. It displays the changes made in the file, with line numbers and a color-coded diff (green for additions, red for deletions). The diff shows that a new paragraph was added to the HTML file.

...	...	@@ -14,5 +14,8 @@
14	14	<p>Ett stycke till - för demoändamål</p>
15	15	<p>Och ett till</p>
16	16	</div>
	17 +	<div>
	18 +	<p>Nytt stycke</p>
	19 +	</div>
17	20	</body>
18	21	</html>
19	22	No newline at end of file

git push

Vi väljer att synka med Git Shell. Kommandot för att skicka iväg filer för synkning (synka uppåt) är: `git push`. Vi gör det.

```
C:\Users\thom_s\Documents\GitHub\git-start [master]> git push
Counting objects: 5, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 339 bytes | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
To https://github.com/mg-teknik/git-start.git
 49fd9f7..b753b09 master -> master
C:\Users\thom_s\Documents\GitHub\git-start [master]>
```

Kontrollera i GitHub for Windows

I GitHub for Windows ser vi att det inte finns något som behöver synkas.

master ▾

History

ändrat i html-fil
1 minute ago by mg-teknik

Lagt till html-fil igen
5 minutes ago by mg-teknik

tagit bort html-fil
6 minutes ago by mg-teknik

html-fil tillagd
14 minutes ago by mg-teknik

Lagt till en rad
17 minutes ago by mg-teknik

Initial commit
23 minutes ago by mg-teknik

ändrat i html-fil

mg-teknik

b753b09

GitHub

Revert

Collapse all

tst.html

3

... ... @@ -14,5 +14,8 @@

14 14 <p>Ett stycke till - för demoändamål</p>

15 15 <p>Och ett till</p>

16 16 </div>

17 17 <div>

18 18 <p>Nytt stycke</p>

19 19 </div>

17 20 </body>

18 21 </html>

19 22 No newline at end of file

GitHub for Windows och GitShell - Thomas Lindbjer

Sida 23

Kontroll på GitHub

Går vi till GitHub ser vi filerna och att de är ändrade bara för ett par minuter sen.

Klicka på `tst.html` för att kontrollera innehållet.

The screenshot shows the GitHub interface for the repository 'mg-teknik / git-start'. At the top, there are buttons for 'Unwatch', 'Star' (0), and 'Fork' (0). Below this is a 'Description' section with a text input field and a 'Website' section with another text input field and 'Save' or 'Cancel' buttons. A summary bar shows '6 commits', '1 branch', '0 releases', and '1 contributor'. Below this, a commit history table lists two commits: 'README.md' (18 minutes ago) and 'tst.html' (2 minutes ago). An orange arrow points to the 'tst.html' entry. The 'tst.html' entry shows the commit message 'ändrat i html-fil'. Below the commit history, the 'README.md' file is displayed, showing the title 'git-start' and the text 'Lägger till en rad.'. On the right side, there is a sidebar with links to 'Code', 'Issues' (0), 'Pull requests' (0), 'Wiki', 'Pulse', 'Graphs', and 'Settings'. At the bottom right, there is a section for 'HTTPS clone URL' with the URL 'https://github.com/mg-teknik/git-start' and buttons for 'Clone in Desktop' and 'Download ZIP'.

mg-teknik / git-start

Unwatch 1 Star 0 Fork 0

Description Website

Short description of this repository Website for this repository (optional) Save or Cancel

6 commits 1 branch 0 releases 1 contributor

branch: master git-start / +

ändrat i html-fil

mg-teknik authored 2 minutes ago latest commit b753b09b02

README.md	Lagt till en rad	18 minutes ago
tst.html	ändrat i html-fil	2 minutes ago

README.md

git-start

Lägger till en rad.

Code

Issues 0

Pull requests 0

Wiki

Pulse

Graphs

Settings

HTTPS clone URL

https://github.com/mg-teknik/git-start

You can clone with HTTPS, SSH, or Subversion.

Clone in Desktop

Download ZIP

Ändring i html-filen

Här ser vi ändringen i html-filen.



The screenshot shows a GitHub repository for 'mg-teknik / git-start'. The file 'tst.html' is selected, showing a commit by 'mg-teknik' 2 minutes ago. The file content is displayed with line numbers 1 to 21. A new paragraph is highlighted with an orange box.

mg-teknik / git-start

branch: master git-start / tst.html

mg-teknik 2 minutes ago ändrat i html-fil

1 contributor

21 lines (21 sloc) 0.398 kb

Raw Blame History

```
1 <!DOCTYPE html>
2 <html lang="sv">
3   <head>
4     <meta charset="utf-8" />
5     <title>Titel</title>
6   </head>
7   <body>
8     <header>
9       <h1>Ett demo-dokument</h1>
10      <p>Utan speciellt innehåll</p>
11    </header>
12    <div class="tst">
13      <p>Inget speciellt innehåll</p>
14      <p>Ett stycke till - för demoändamål</p>
15      <p>Och ett till</p>
16    </div>
17    <div>
18      <p>Nytt stycke</p>
19    </div>
20  </body>
21 </html>
```

Översikt över kommando för git

En gång till: gits olika nivåer.

Git är alltså en databas som håller reda på alla ändringar vi ber den hålla reda på - inte hela filer. Det är en av anledningarna till att den är så effektiv.

1. Den lokala mappen där vi har våra filer
2. Här lägger vi till de filer som är klara att läggas till i gits databas. Git håller givetvis ordning på dessa filer också. Om vi ändrar dessa filer är de "modifierade" och behöver läggas till igen (för att kunna commitas till databasen).
3. Först när vi lägger till filerna med commit hamnar alla som finns i staging area i gits databas över alla ändringar.
4. I Remote repo finns filerna när vi lägger över dem till den tjänst vi använder på nätet, t.ex. GitHub eller Bitbucket. Det är även härifrån man kan hämta eller kлона repot.

