

# git - Der einfache Einstieg

eine einfache Anleitung, um git zu lernen. Kein Schnick-Schnack ;)

Twittern

von Roger Dudler

Dank an @tfnico, @fhd und Namics

ng in english, español, français, indonesian, italiano, nederlands, polski, português, pycc

မြန်မာ, 日本語, 中文, 한국어

日本語, 中文, 한국어

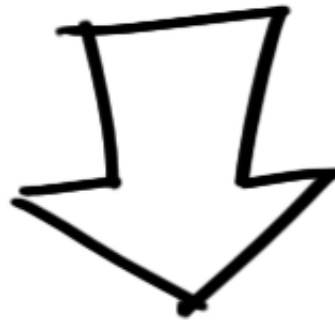
Feedback auf github



**Are You a Front-End Developer?**  
by Roger Dudler, Author of the Git Simple Guide

**Try Frontify**

Now Free with  
**Github** Integration!



# installation

git für OS X herunterladen

git für Windows herunterladen

git für Linux herunterladen

# neues repository erstellen

erstelle ein neues Verzeichnis, öffne es und führe

```
git init
```

aus, um ein neues git-Repository anzulegen.

# ein repository auschecken

erstelle eine Arbeitskopie, indem du folgenden Befehl ausführst:

```
git clone /pfad/zum/repository
```

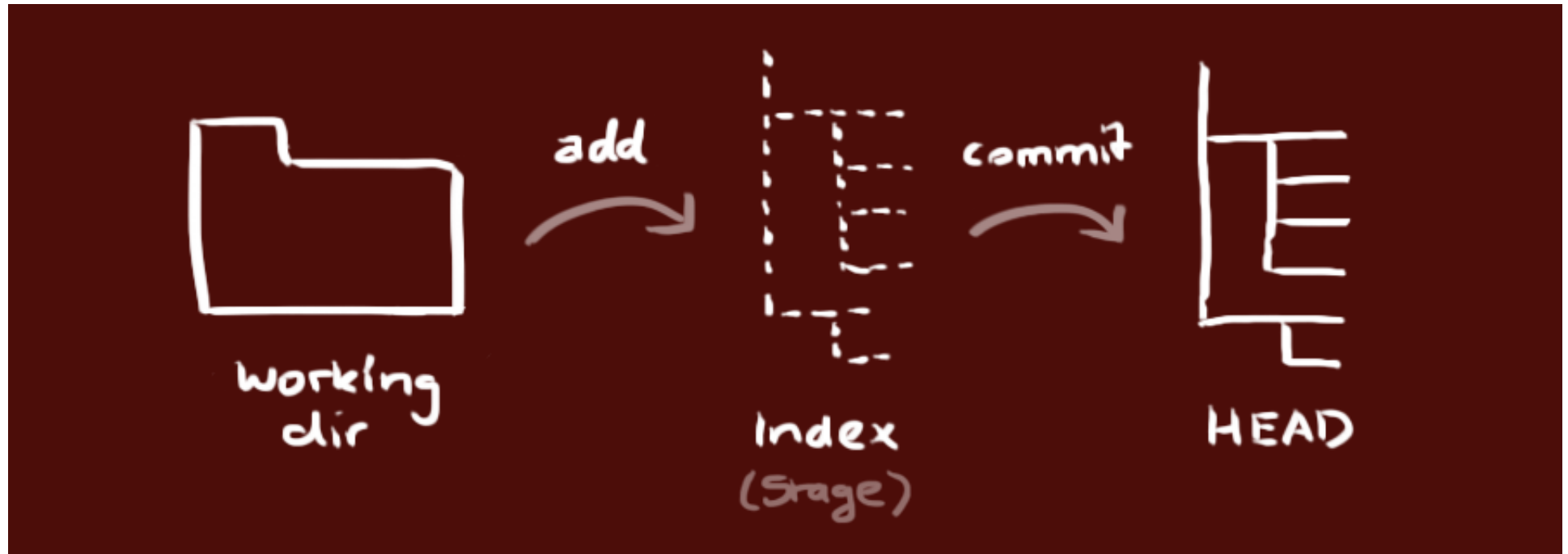
Falls du ein entferntes Repository verwendest, benutze:

```
git clone benutzername@host:/pfad/zum/repository
```

# workflow

Dein lokales Repository besteht aus drei "Instanzen", die von git verwaltet werden. Die erste ist deine **Arbeitskopie**, welche die echten Dateien enthält. Die zweite ist der **Index**, welcher als Zwischenstufe agiert und zu guter Letzt noch der **HEAD**, der auf

deinen letzten Commit zeigt.



# add & commit

Du kannst Änderungen vorschlagen (zum **Index** hinzufügen) mit

```
git add <dateiname>
```

```
git add *
```

Das ist der erste Schritt im git workflow, du bestätigst deine

Änderungen mit:

```
git commit -m "Commit-Nachricht"
```

Jetzt befindet sich die Änderung im **HEAD**, aber noch nicht im

entfernten Repository.

# änderungen hochladen

Die Änderungen sind jetzt im **HEAD** deines lokalen Repositories. Um die Änderungen an dein entferntes Repository zu senden, führe:

```
git push origin master
```

aus. Du kannst *master* auch mit einem beliebigen anderen Branch



ersetzen, mehr über Branches erfährst du später.

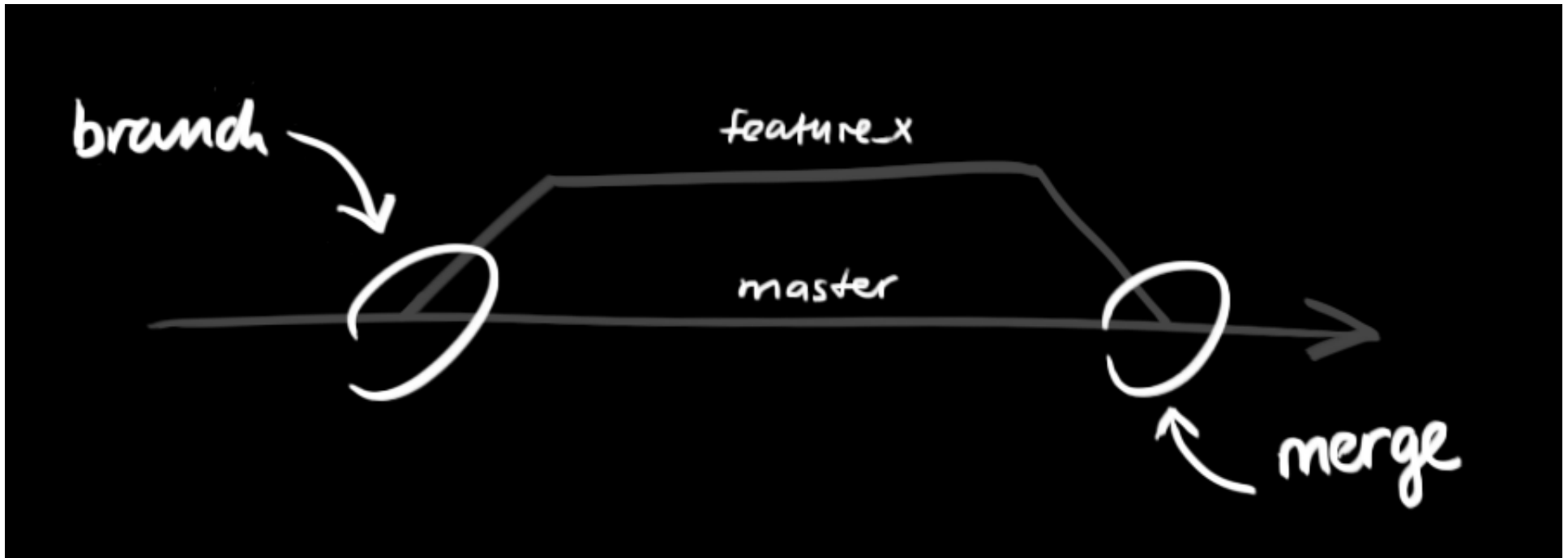
Wenn du dein lokales Repository nicht von einem entfernten geklont hast und du diese aber mit einem anderen Repository verbinden möchtest, musst du dieses mit

```
git remote add origin <server>
```

hinzufügen. Jetzt bist du bereit, deine Änderungen hochzuladen

# branching

Branches werden benutzt, um verschiedene Funktionen isoliert voneinander zu entwickeln. Der *master*-Branch ist der "Standard"-Branch, wenn du ein neues Repository erstellst. Du solltest aber für die Entwicklung andere Branches verwenden und diese dann in den Master-Branch zusammenführen (mergen). Auch das lernst du später.



Erstelle einen neuen Branch mit dem Namen "feature\_x" und wechsle zu diesem:

```
git checkout -b feature_x
```

Um zum Master zurück zu wechseln:

```
git checkout master
```

Und um den eben erstellten Branch wieder zu löschen:

```
git branch -d feature_x
```

Ein Branch ist *nicht für andere verfügbar*, bis du diesen in dein entferntes Repository hochlädst:

```
git push origin <branch>
```

# update & merge

Um dein lokales Repository mit den neuesten Änderungen zu aktualisieren, verwende:

```
git pull
```

in deiner Arbeitskopie, um die Änderungen erst *herunterzuladen (fetch)* und dann mit deinem Stand *zusammenzuführen (merge)*.

Wenn du einen anderen Branch mit deinem aktuellen (z.B. master) zusammenführen willst, benutze:

```
git merge <branch>
```

In beiden Fällen versucht git die Änderungen automatisch zusammenzuführen. Unglücklicherweise ist dies nicht immer möglich und endet in *Konflikten*. Du bist verantwortlich, diese *Konflikte* durch

manuelles Editieren der betroffenen Dateien zu lösen. Bist du damit fertig, musst du das git mit folgendem Befehl mitteilen:

```
git add <dateiname>
```

Bevor du Änderungen zusammenführst, kannst du dir die Differenzen auch anschauen:

```
git diff <quell_branch> <ziel_branch>
```

# tagging

Es wird empfohlen, für Software Releasestags zu verwenden. Dies ist ein bekanntes Konzept, das es schon mit SVN gab. Du kannst einen neuen Tag namens *1.0.0* mit folgendem Befehl erstellen:

```
git tag 1.0.0 1b2e1d63ff
```

*1b2e1d63ff* steht für die ersten 10 Zeichen der Commit-Id, die du mit deinem Tag referenzieren möchtest. Du erhältst die Liste der Commit-IDs mit:

```
git log
```

Du kannst auch weniger Zeichen verwenden, es muss einfach eindeutig sein.

# änderungen rückgängig machen

Falls du mal etwas falsch machst (was natürlich nie passiert ;) ) kannst

du die lokalen Änderungen mit:

```
git checkout -- <filename>
```

auf den letzten Stand im HEAD zurücksetzen. Änderungen, die du bereits zum Index hinzugefügt hast, bleiben bestehen.



Wenn du aber deine lokalen Änderungen komplett entfernen möchtest, holst du dir den letzten Stand vom entfernten Repository mit folgenden

Befehlen:

```
git fetch origin
```

```
git reset --hard origin/master
```

# nützliche tricks

Eingebaute git-GUI:

`gitk`

Farbige Konsolenausgabe:

`git config color.ui true`

Eine Zeile pro Commit in der Logausgabe:

`git config format.pretty oneline`

Interaktives Hinzufügen von Änderungen:

`git add -i`

# links

## grafische clients

GitX (L) (OS X, Open Source)

Tower (OS X)

Source Tree (OS X, kostenlos)

GitHub for Mac (OS X, kostenlos)

GitBox (OS X)

## anleitungen

Git Community Book

Pro Git

Think like a git

GitHub Help

# kommentare

9 Comments

git - the simple guide

 Login ▾

 Recommend 2

 Share

Sort by Newest ▾



Join the discussion...



**DAU** · 2 months ago

that was very helpful thanks alot !!!

^ | ▾ · Reply · Share ›



**mik** · 3 months ago

Thanks, gute Hilfe .... hatte nach X Versuchen immer noch kein Plan. Das half ;-)

^ | ▾ · Reply · Share ›



**Martin Lehmann** · 4 months ago

Herzlichen Dank, passt genau für einen SVN->GIT-Wechsler!

^ | v · Reply · Share ›



**Steffen** · 4 months ago

Danke!!! :)

^ | v · Reply · Share ›



**Steffen** → Steffen · 4 months ago

sehr gut für Anfänger!

^ | v · Reply · Share ›



**inka** · 4 months ago

Danke! Ich hab echt keine Ahnung, wie Du Dich in DAUs wie mich so reinversetzen konntest, dass diese Anleitung zustande kam, aber herzlichen Glückwunsch, alle Kollegen sind dran gescheitert. :D

^ | v · Reply · Share ›



**Vincent** · 5 months ago

Sehr gut für den Einstieg in GIT. Jedem Anfänger empfehle ich diese Seite.

^ | v · Reply · Share ›



**Marius** · 5 months ago

Vielen Dank!!

^ | v · Reply · Share ›



**fabi** · 6 months ago

wow, i guess this guid all i ever wanted. thank u

^ | v · Reply · Share ›

## git - Der einfache Einstieg - kein Schnick-schnack!

36 comments • 3 years ago

**u** — Unter Änderungen hochladen:  
"Wenn du dein lokales Repository nicht  
von einem entfernten geklont hast ...

## git - hướng dẫn đơn giản

22 comments • 2 years ago

**ubuntu** — Bài viết của bạn rất hay, hy  
vọng có nhiều bài viết hay nữa

## git - la guida tascabile - niente di complicato!

43 comments • 3 years ago

**neokree** — no, e' utile per chi sa cosa sia  
il controllo di versione, i repo, i commit,  
i branch e le pull request. Se devi ...

## git - ก้าวแรกสู่สังเวียน

5 comments • a year ago

**Prapath Nui Suayroop** — ขอบคุณครับ คง  
ต้องอ่านอีกหลาย ๆ ที่ แต่เริ่มเข้าใจมากขึ้น  
กว่าตอนอ่าน Text แล้วละครับ

---

 [Subscribe](#)

 [Add Disqus to your site](#)

 [Privacy](#)