

Git kennenlernen

Anfänger

Erste Schritte

Zusammenarbeit

Synchronisierung

[git remote](#)[git fetch](#)[git push](#)[git pull](#)

Einen Pull Request erstellen

Arbeiten mit Branches

Workflows vergleichen

Migration zu Git

Tipps für



## Git-Synchronisierung

### **git remote / git fetch / git push / git pull**

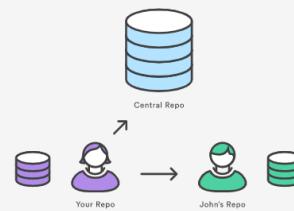
SVN nutzt ein einziges zentralisiertes Repository als Kommunikationsschnittstelle für Entwickler. Die Zusammenarbeit erfolgt dabei über Changesets, die zwischen den Arbeitskopien der Entwickler und dem zentralen Repository ausgetauscht werden. Genauso darin liegt der Unterschied zum verteilten Zusammenarbeitsmodell von Git. Bei Git erhält jeder Entwickler eine eigene Kopie des Repositorys, die einen eigenen lokalen Verlauf und eine eigene lokale Branch-Struktur aufweist. In der Regel müssen Benutzer eine Reihe aufeinanderfolgender Commits veröffentlichen, kein einzelnes Changeset. Mit Git committest du keine Changesets von einer Arbeitskopie zum zentralen Repository, sondern kannst ganze Branches zwischen Repositorys teilen.

Der Befehl `git remote` ist Teil eines breiten Systems, das für die Synchronisierung von Änderungen zuständig ist. Einträge, die über `git remote` registriert werden, werden zusammen mit den Befehlen `git fetch`, `git push` und `git pull` verwendet. Diese Befehle haben alle ihre eigenen Synchronisierungsaufgaben, die in den entsprechenden Links weiter ausgeführt werden.

## git remote

Mit dem Befehl `git remote` kannst du Verbindungen mit anderen Repositorys erstellen, abrufen und löschen. Dabei solltest du dir Remote-Verbindungen wie Lesezeichen vorstellen, weniger als direkte Links zu anderen Repositorys. Sie erlauben keinen Echtzeitzugriff auf die Repositorys. Vielmehr sind sie praktische Kurznamen, die du statt langer und komplizierter URLs zum Referenzieren verwenden kannst.

Das folgende Diagramm zeigt zum Beispiel zwei Remote-Verbindungen von deinem Repository zum zentralen Repository und zum Repository eines anderen Entwicklers. Statt Repository-Referenzen über deren vollständige URLs herzustellen, kannst du das Original einfach weitergeben und dein Kollege John erstellt Verknüpfungen zu anderen Git-Befehlen.



## Übersicht über die Verwendung von Git Remote

Der Befehl `git remote` ist im Wesentlichen eine Schnittstelle für das Management einer Liste von Remote-Einträgen, die in der Datei `./.git/config` des Repositorys gespeichert sind. Die folgenden Befehle werden zur Anzeige des aktuellen Status der Remote-Liste verwendet.

## Anzeigen von git remote-

# Konfigurationen

```
git remote
```

Führt deine Remote-Verbindungen zu anderen Repositorys auf

```
git remote -v
```

Wie der Befehl oben, aber einschließlich URL jeder Verbindung

## Erstellen und Ändern von git remote- Konfigurationen

Der Befehl `git remote` ist außerdem eine praktische Methode bzw. ein Hilfsprogramm für das Ändern der Datei `./.git/config` eines Repositorys. Mit den unten aufgeführten Befehlen kannst du Verbindungen mit anderen Repositorys verwalten. Die folgenden Befehle führen Änderungen an der Datei `./.git/config` durch. Dasselbe Ergebnis kann auch erzielt werden, indem du die Datei `./.git/config` direkt in einem Text-Editor bearbeitest.

```
git remote add <name> <url>
```

Mit diesem Befehl erstellst du eine neue Verbindung mit einem Remote-Repository. Sobald du das Remote-Repository hinzugefügt hast, kannst du in anderen Git-Befehlen den Wert aus dem Argument "<name>" als Kurzform des Werts aus dem Argument "<url>" verwenden.

```
git remote rm <name>
```

Mit diesem Befehl entfernst du die Verbindung mit dem im Argument <name> angegebenen Remote-Repository.

```
git remote rename <old-name> <new-name>
```

Mit diesem Befehl benennst du eine Remote-Verbindung von dem als "<old-name>" angegebenen Wert in den als "<new-name>" angegebenen Wert um.

## Über git remote

Git wurde so konzipiert, dass jeder Entwickler in einer völlig isolierten Entwicklungsumgebung arbeiten kann. Das bedeutet, dass Informationen nicht automatisch zwischen den Repositorys ausgetauscht werden. Stattdessen müssen Entwickler Upstream-Commits manuell in ihr lokales Repository pullen oder ihre lokalen Commits manuell zurück in das zentrale Repository pushen. Der Befehl `git remote` ist einfach ein einfacherer Weg, um URLs an diese "Freigabebefehle" weiterzugeben.

## Die Remote-Verknüpfung origin

Wenn du ein Repository mit `git clone` klonst, wird automatisch eine Remote-Verbindung namens "origin" erstellt, die auf das geklonte Repository zurückverweist. Das ist hilfreich für Entwickler, die eine lokale Kopie eines zentralen Repositorys erstellen, da sie auf diese Weise ganz einfach Upstream-Änderungen pullen oder lokale Commits veröffentlichen können. Dieses Verhalten ist auch der Grund, warum das zentrale Repository in den meisten Git-Projekten "origin" heißt.

## Repository-URLs

Git bietet verschiedene Möglichkeiten, Verweise zu einem Remote-Repository herzustellen. Über HTTP- und SSH-Protokollen kann man am einfachsten auf ein Remote-Repository zugreifen. HTTP bietet eine einfache Möglichkeit für einen anonymen schreibgeschützten Zugriff auf ein Repository. Ein Beispiel:

```
http://host/path/to/repo.git
```

Allgemein ist es jedoch nicht möglich, Commits an eine HTTP-Adresse zu pushen (wobei du anonyme Pushes sowie nicht erlauben wolltest). Für Lese- und Schreibzugriff solltest du stattdessen SSH verwenden:

```
ssh://user@host/path/to/repo.git
```

Du benötigst ein gültiges SSH-Konto auf der Hostmaschine. Im Übrigen unterstützt Git einen authentifizierten Zugriff über SSH ohne vorherige Einrichtung. Moderne, sichere Hosting-Lösungen von Drittanbietern wie Bitbucket.com stellen diese URLs für dich bereit.

## git remote-Befehle

Der Befehl `git remote` ist einer der vielen Git-Befehle, die durch zusätzlich angehängte "Unterbefehle" ergänzt werden können. Im Folgenden sehen wir uns häufig genutzte `git remote`-Unterbefehle an.

```
ADD <NAME> <URL>
```

Fügt einen Eintrag zu `./.git/config` für das Remote-Repository namens `<name>` unter der Repository-URL `<url>` hinzu.

Akzeptiert die Option `-t`, die sofort nach dem Erstellen des Remote-Eintrags `git fetch <name>` ausführt.

Akzeptiert die Option `-tags`, die umgehend `git fetch <name>` ausführt und jeden Tag vom Remote-Repository importiert.

```
RENAME <OLD> <NEW>
```

Aktualisiert in `./.git/config` die Umbenennung des Eintrags `<OLD>` in `<NEW>`. Alle remote verfolgten Branches und Konfigurationseinstellungen für das Remote-Repository werden aktualisiert.

```
REMOVE or RM <NAME>
```

Ändert `./.git/config` und entfernt das Remote-Repository namens `<NAME>`. Alle remote verfolgten Branches und Konfigurationseinstellungen für das Remote-Repository werden entfernt.

```
GET-URL <NAME>
```

Gibt die URLs eines Remote-Eintrags aus.

Akzeptiert `--push`, Push-URLs werden statt Fetch-URLs abgefragt.

Mit `--all` werden alle URLs für das Remote-Repository aufgeführt.

```
SHOW <NAME>
```

Gibt allgemeine Informationen über das Remote-Repository `<NAME>` aus.

```
PRUNE <NAME>
```

Löscht alle lokalen Branches von `<NAME>`, die nicht im Remote-Repository vorhanden sind.

Akzeptiert die Option `--dry-run`, die auflistet, welche Branches gelöscht werden sollen, ohne den tatsächlichen Löschvorgang auszuführen.

## Beispiele zu git remote

Neben einer Verbindung mit "origin" sind auch Verbindungen mit den Repositorys deiner Teamkollegen sehr praktisch. Wenn dein Kollege John beispielsweise ein öffentlich zugängliches Repository

unter dev.example.com/john.git pflegt, könntest du wie folgt eine entsprechende Verbindung einrichten:

```
git remote add john http://dev.example.com/john.git
```

Mit einem solchen Zugriff auf die Repositorys einzelner Entwickler könnt ihr außerhalb der zentralen Repositorys zusammenarbeiten. Das kann für kleine Teams bei einem Großprojekt sehr nützlich sein.

## Anzeige deiner Remote-Repositorys

Standardmäßig listet der Befehl `git remote` zuvor gespeicherte Remote-Verbindungen zu anderen Repositorys auf. Dadurch wird eine einzeilige Ausgabe erzeugt, in der die Namen der "Lesezeichennamen" der Remote-Repositorys aufgeführt werden.

```
$ git remote
origin
upstream
other_users_repo
```

Durch das Aufrufen von `git remote` mit der Option `-v` wird eine Liste von Repository-Namen, für die ein Lesezeichen gesetzt wurde, und ihre entsprechenden Repository-URLs gedruckt. Die Option `-v` steht für "verbose" (ausführlich). Im Folgenden siehst du ein Beispiel für eine ausführliche `git remote`-Ausgabe.

```
git remote -v
origin git@bitbucket.org:origin_user/reponame.git (f
origin git@bitbucket.org:origin_user/reponame.git (pu
upstream https://bitbucket.org/upstream_user/repos
upstream https://bitbucket.org/upstream_user/repos
other users repo https://bitbucket.org/other users
other users repo https://bitbucket.org/other users
```

## Hinzufügen von Remote-Repositorys

Der Befehl `git remote add` erstellt einen neuen Eintrag einer Verbindung zu einem Remote-Repository. Nach dem Hinzufügen eines Remote-Repositorys kannst du in anderen Git-Befehlen `<name>` als eine praktische Verknüpfung für `<url>` verwenden. Weitere Informationen zur akzeptierten URL-Syntax findest du unten im Abschnitt "Repository-URLs". `git remote add` erstellt einen neuen Eintrag in der Datei `./.git/config` des Repositorys. Diese Konfigurationsdatei sieht beispielsweise folgendermaßen aus:

```
$ git remote add fake_test https://bitbucket.com/upst
url = https://bitbucket.com/upstream_user/reponame.
fetch = +refs/heads/*:refs/remotes/remote_test/*
```

## Überprüfen eines Remote-Repositorys

Der Unterbefehl "show" kann an `git remote` angehängt werden, um eine detaillierte Ausgabe zur Konfiguration eines Remote-Repositorys zu erhalten. Diese Ausgabe enthält eine Liste von Branches, die dem Remote-Repository zugeordnet sind, sowie die angefügten Endpunkte für Fetch- und Push-Vorgänge.

```
git remote show upstream
* remote upstream
  Fetch URL: https://bitbucket.com/upstream_user/repo
  Push URL: https://bitbucket.com/upstream_user/repo
  HEAD branch: master
  Remote branches:
    master tracked
    simd-deprecated tracked
    tutorial tracked
  Local ref configured for 'git push':
    master pushes to master (fast-forwardable)
```

## Fetch- und Pull-Vorgänge von Git-Remotes

Sobald ein Remote-Eintrag mithilfe des Befehls `git remote`

konfiguriert wurde, kann der Remote-Name in anderen Git-Befehlen als Argument genutzt werden, um mit dem Remote-Repository zu kommunizieren. Sowohl `git fetch` als auch `git pull` kann für das Lesen eines Remote-Repositories verwendet werden. Beide Befehle lösen unterschiedliche Vorgänge aus, die in ihren jeweiligen Links im Detail erläutert werden.

## Pushen zu Git-Remotes

Der Befehl `git push` wird zum Schreiben auf ein Remote-Repository verwendet.

```
git push <remote-name> <branch-name>
```

In diesem Beispiel wird der lokale Zustand von `<branch-name>` auf das mit `<remote-name>` angegebene Remote-Repository hochgeladen.

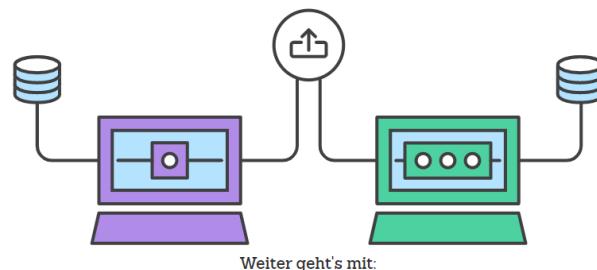
## Umbenennen und Entfernen von Remotes

```
git remote rename <old-name> <new-name>
```

Der Befehl `git remote` erklärt sich von selbst. Wenn dieser Befehl ausgeführt wird, wird eine Remote-Verbindung von `<old-name>` zu `<new-name>` umbenannt. Außerdem wird der Eintrag für das Remote-Repository auch im Verzeichnis `./.git/config` umbenannt.

```
git remote rm <name>
```

Der Befehl `git remote rm` entfernt die Verbindung zum mit dem `<name>`-Parameter angegebenen Remote-Repository. Machen wir zu Demonstrationszwecken einmal das im letzten Beispiel hinzugefügte Remote-Repository rückgängig. Wenn wir `git remote rm remote_test` ausführen und dann die Inhalte von `./.git/config` ansehen, stellen wir fest, dass der Eintrag `[remote "remote_test"]` entfernt wurde.



Weiter geht's mit:

**git fetch**

[NÄCHSTES TUTORIAL BEGINNEN](#)

Entwickelt von

Bitbucket

Empfehl uns weiter



Interesse an künftigen Artikeln?

Enter Your Email For Git News

Website gehostet von

ATTLASIAN