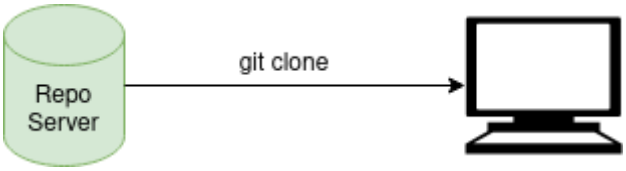
# Geklonte Repositorys

Zunächst ist ein Git-Repo wie wir gesehen haben, eine vollkommen lokale Angelegenheit - alle wichtigen Infos und die Snapshots werden im .git-Verzeichnis gespeichert.

Um besser zusammenarbeiten zu können ist es möglich, ein Repo über entsprechende Mechanismen anderen Menschen zur Verfügung zu stellen, z.B. via SSH oder https. Diese Veröffentlichung muss nicht unbedingt öffentlich geschehen, sondern kann durchaus z.B. mit einem Passwort geschützt werden, so dass nur Mitglieder einer bestimmten Gruppe Zugriff auf das Repo haben. Außerdem kann unterschieden werden zwischen lesendem Zugriff und schreibendem Zugriff.

## Klonen eines Repositorys

Ein so veröffentlichtes Repo kann man "klonen". Der Befehl lautet z.B. `git clone https://codeberg.org/opyale/GitNex.git`. Die Adresse, von der das Repo geklont wird, heißt `origin`.




```
git clone https://codeberg.org/ironiemix/uebungsserver.git
Klone nach 'uebungsserver' ...
remote: Enumerating objects: 163, done.
remote: Counting objects: 100% (163/163), done.
remote: Compressing objects: 100% (126/126), done.
remote: Total 163 (delta 52), reused 0 (delta 0)
Empfange Objekte: 100% (163/163), 23.05 KiB | 1.92 MiB/s, fertig.
Löse Unterschiede auf: 100% (52/52), fertig.
```

Nun existiert eine vollständige lokale Kopie auf dem lokalen Rechner, die alle Commits des ursprünglichen Repos nachverfolgbar enthält:

```
[frank@rita uebungsserver]$ git log --graph --pretty=format:'%Cred%h%Creset -%C(yellow)%d%Creset %s %Cgreen(%cr) %C(bold blue)<%an>%Creset' --abbrev-com
* c73c1d2 - (HEAD -> master, origin/master, origin/HEAD) Fix phpmyadmin not working due to open_basedir restrictions (vor 3 Tagen) <Frank Schiebel>
* 6f67001 - Abfrage nach Domain in install.sh (vor 3 Tagen) <Frank Schiebel>
* 3660798 - Interactive PS generation, phpmyadmin from backports (vor 6 Monaten) <Frank Schiebel>
* 68d1cb8 - Install pwgen first, then use it (vor 6 Monaten) <Frank Schiebel>
* 8bb5c79 - Autoinstall (vor 6 Monaten) <Frank Schiebel>
* 75bd4d2 - Apache landing page added. First Version of autoinstall script (vor 6 Monaten) <root>
* 9b2db54 - Import the repository keys first... (vor 6 Monaten) <root>
* 32bd030 - Reenabled SSL config when necessary (vor 6 Monaten) <Frank Schiebel>
* 6338483 - Anpassungen für den Raspi (vor 6 Monaten) <Frank Schiebel>
* 6ceaabc - SSL Config (vor 6 Monaten) <root>
* b78b58c - Phpmyadmin test 2 (vor 6 Monaten) <Frank Schiebel>
* 92fe915 - Test phpmyadmin (vor 6 Monaten) <Frank Schiebel>
* fae5e0d - install ansible from backports due to bug in mysql_user in 2.7 (vor 6 Monaten) <Frank Schiebel>
* 4377fd1 - First commit, needs further testing (vor 6 Monaten) <Frank Schiebel>
```

## Änderung auf den Server zurückkopieren

Nun kann man mit dem Repo lokal ganz normal arbeiten, der Unterschied zum "lokalen" Repo ist,

dass dieses Repository weiss, woher es kommt, und das ermöglich es auch, Änderungen wieder auf den entfernten Server zurückzu"pushen". Der dazu verwendete Befehl lautet `git push`.

Zunächst bearbeitet man lokal Dateien im Repo und erzeugt einen (oderer mehrere Commits) :

```
[frank@rita uebungsserver]$ vi uebungsserver.sh
[frank@rita uebungsserver]$ git add uebungsserver.sh
[frank@rita uebungsserver]$ git commit -m "Lizenz und Hinweise"
[master cb35baf] Lizenz und Hinweise
 1 file changed, 8 insertions(+)
[frank@rita uebungsserver]$ git lg
* cb35baf - (HEAD -> master) Lizenz und Hinweise (vor 5 Sekunden) <Frank Schiebel>
* c73c1d2 - (origin/master, origin/HEAD) Fix phpmyadmin not working due to open_basec
* 6f67001 - Abfrage nach Domain in install.sh (vor 3 Tagen) <Frank Schiebel>
* 3660798 - Interactive PS generation, phpmyadmin from backports (vor 6 Monaten) <Fra
* 68d1cb8 - Install pwgen first, then use it (vor 6 Monaten) <Frank Schiebel>
* 8bb5c79 - Autoinstall (vor 6 Monaten) <Frank Schiebel>
* 75bd4d2 - Apache landing page added. First Version of autoinstall script (vor 6 Mor
* 9b2db54 - Import the repository keys first... (vor 6 Monaten) <root>
```

Mit dem Befehl `git push` kann man die Änderungen nun zum "origin" zurückpushen:

```
[frank@rita uebungsserver]$ git push
Username for 'https://codeberg.org': [redacted]
Password for 'https://[redacted]@codeberg.org':
Objekte aufzählen: 5, fertig.
Zähle Objekte: 100% (5/5), fertig.
Delta-Kompression verwendet bis zu 4 Threads.
Komprimiere Objekte: 100% (3/3), fertig.
Schreibe Objekte: 100% (3/3), 469 Bytes | 469.00 KiB/s, fertig.
Gesamt 3 (Delta 1), Wiederverwendet 0 (Delta 0), Pack wiederverwendet 0
remote: . Processing 1 references
remote: Processed 1 references in total
To https://codeberg.org/ironiemix/uebungsserver.git
   c73c1d2..cb35baf  master -> master
```

Damit landen die Änderungen auf dem ursprünglichen Server.

## Aktuell bleiben mit "pull"

Um Probleme beim "pushen" von Änderungen zu vermeiden - besonders wenn man in einem Team an einem Repo arbeitet, oder wenn man das Repo auf mehreren Rechnern gleichzeitig bearbeitet, muss man seine lokale Arbeitskopie aktuell halten. Das geschieht mit dem Befehl `git pull`.

Man sollte vor Arbeitsbeginn pullen und bevor man versucht zu pushen. wenn Konflikte auftreten, weil zwischenzeitlich Änderungen auf das zentrale Repository gepusht wurden, kann man diese zunächst lösen und dann das Ergebnis pushen. Wie man Konflikte löst betrachten wir gesondert.