BAB.Entwicklung – Git

Git-Versionskontrolle: Workflows, Branches und Best Practices

Git Schulung

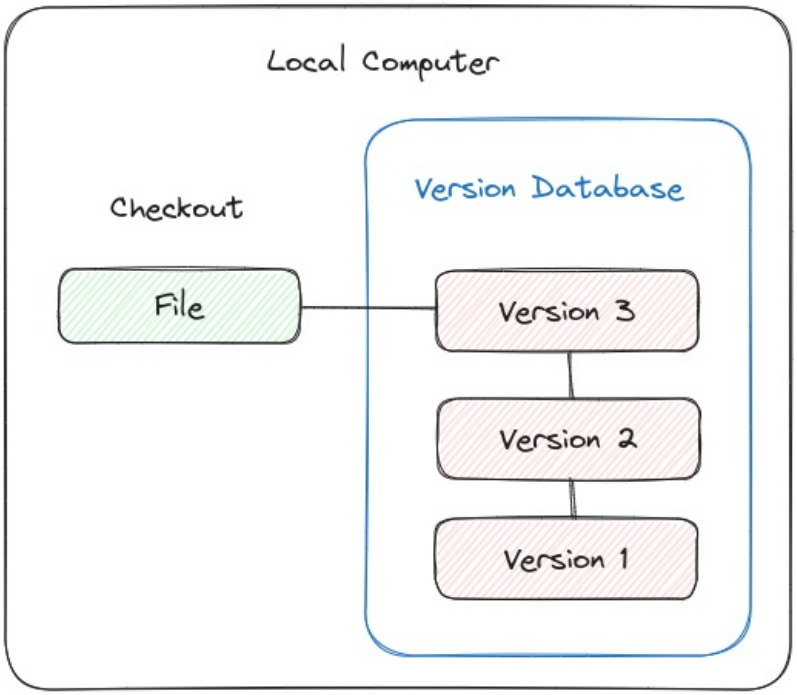
Git Schulung

Einleitung / Geschichte

Git ist ein Distributed Version Control System (DVCS), welches von Linus Torvalds (Erfinder und Maintainer des Linux Kernels) im Jahr 2005 entwickelt wurde.

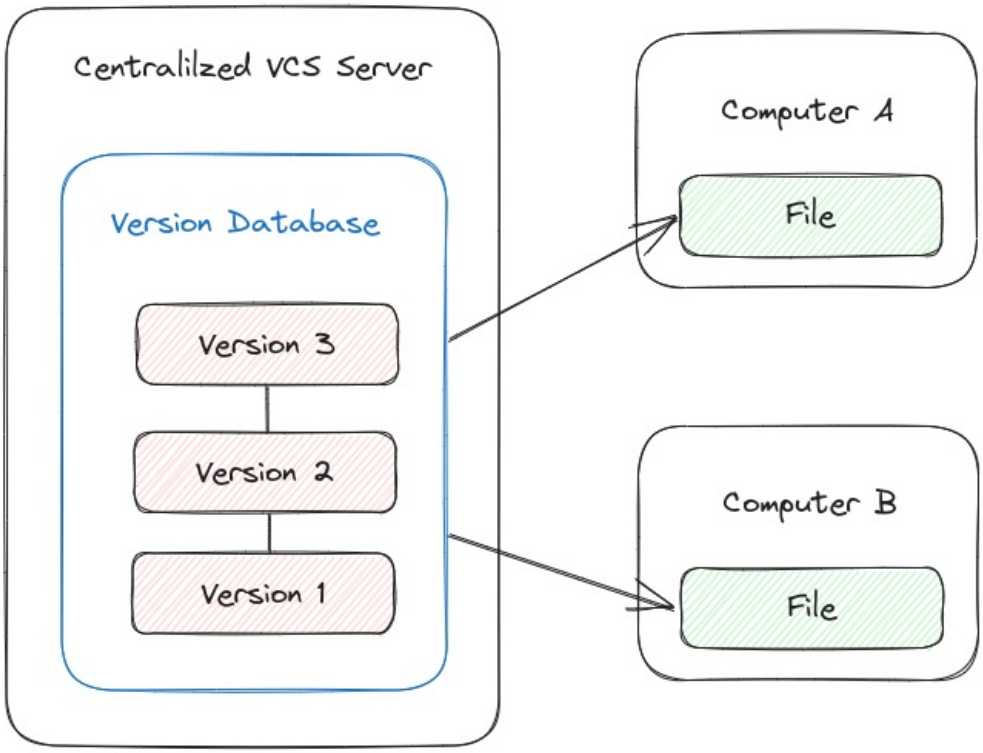
Local VCS

Existiert nur auf einem Computer



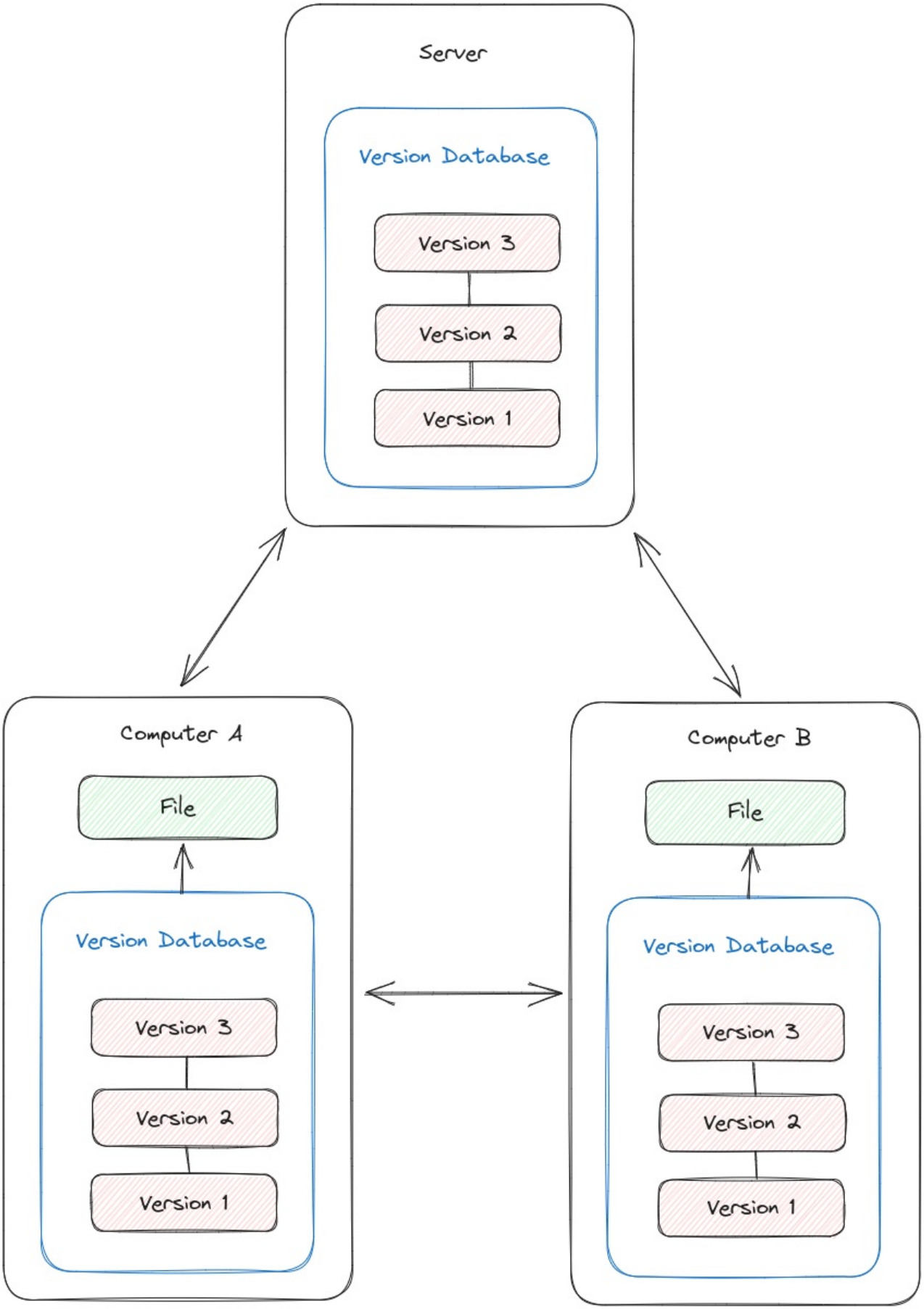
Centralized VCS

Versionsdatenbank wird auf einem zentralen Server gespeichert. Ist dieser nicht erreichbar, kann niemand auf die verschiedenen Versionen zugreifen.



Distributed VCS

Versionsdatenbank wird komplett synchronisiert. Internetverbindung wird nur zum Übertragen von Änderungen benötigt.



Installation Guide

Installation

1. `Win + R` drücken.
2. Den folgenden Befehl kopieren, einfügen und bestätigen.

```
winget install --id Git.Git -e --source winget
```

3. Anweisungen auf dem Bildschirm folgen und bestätigen.

Oder von der Offiziellen Seite herunterladen und installieren.

Die Git Konsole kann nun mit Rechtsklick in einem Ordner und `Open Git Bash here` geöffnet werden. Alle nachfolgenden Befehle beziehen sich auf die `Git Bash Konsole`.

▼ **Falls nicht vorhanden einen Beliebigen Editor installieren**

z.B. Notepad++:

```
winget install -e --id Notepad++.Notepad++
```

Git konfigurieren

```
git config --global user.name "John Doe"
git config --global user.email johndoe@example.com
git config --global init.defaultBranch main
git config --global core.editor nano
```

Wobei der Editor auch auf einen beliebigen anderen Editor gesetzt werden kann.

Usage Guide

Initialisierung eines Repositories

Anlegen eines neuen Repositories:

```
git init
git add .
git commit -m "Initial Commit"
```

Ein neues remote Repository hinzufügen:

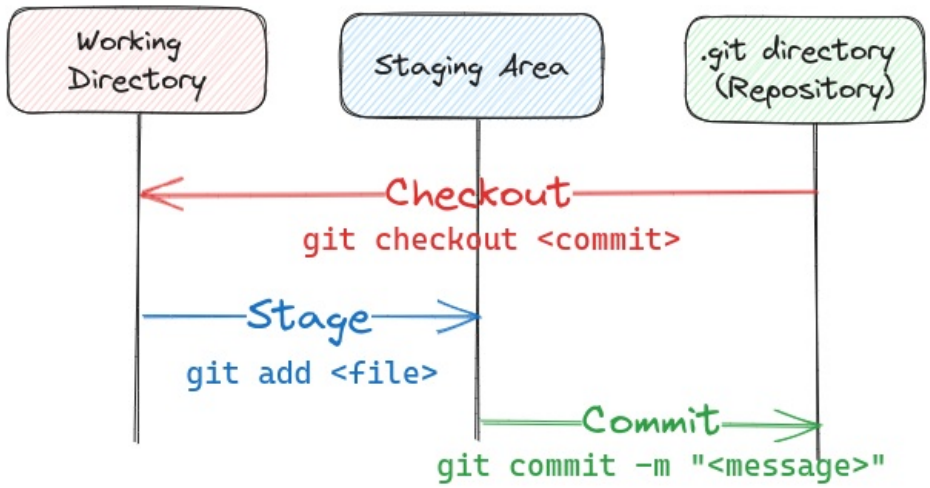
```
git remote add origin https://git.agrarforschung.at/playground/my-first-project.git
git push -u origin main
```

Oder ein bereits existierendes Repository klonen:

```
git clone https://git.agrarforschung.at/playground/my-first-project.git
cd my-first-project
```

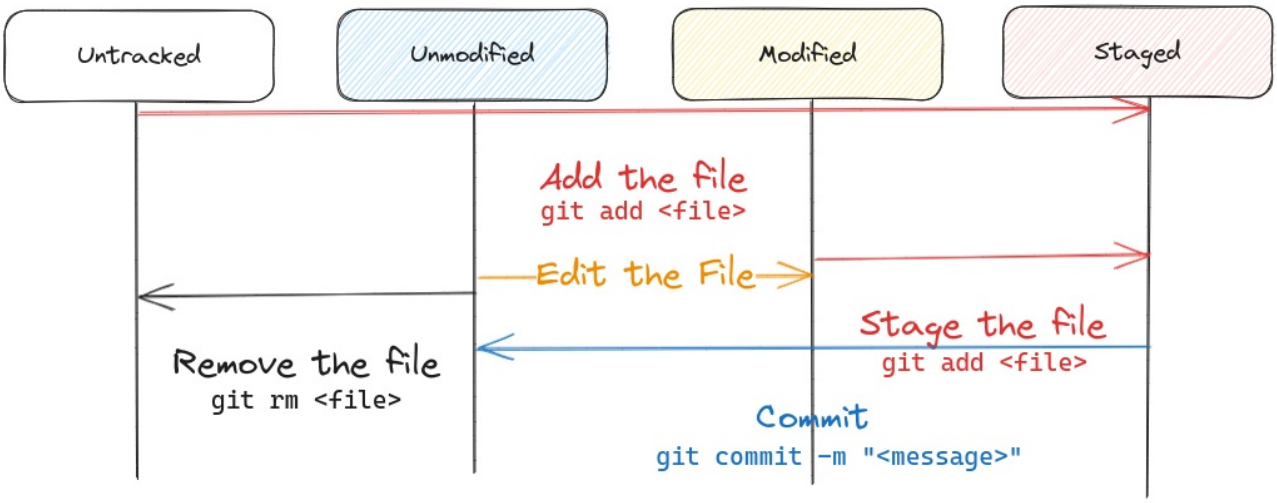
Git States

- Das Working Directory stellt die Dateien dar, wie sie sich aktuell auf der Festplatte befinden.
- Die Staging Area dient zum Vormerken für Änderungen.
- Das Repository beinhaltet alle vorgegangenen Änderungen.



File States

State	Beschreibung
Untracked	Datei wird derzeit nicht vom VCS getrackt
Unmodified	Datei wird vom VCS getrackt, aber wurde nicht verändert
Modified	Datei wird vom VCS getrackt, allerdings gibt es Änderungen
Staged	Datei ist zum Committen vorgemerkt



Informationen zum Repository erhalten

Zustand der Dateien abfragen:

```
git status
```

Verlauf ansehen:

```
git log
```

Änderungen ansehen:

```
git diff
```

Änderungen einchecken

Änderungen stagen:

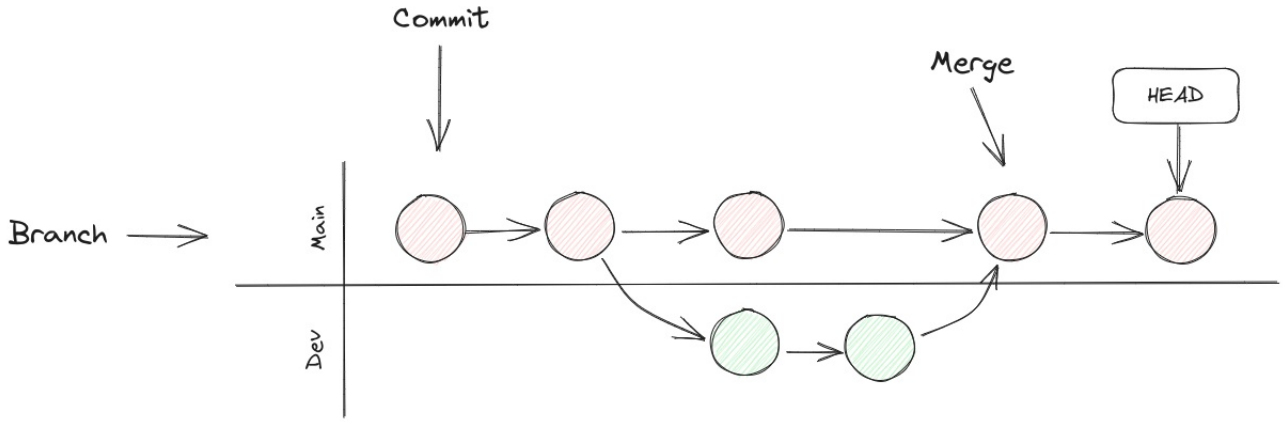
```
git add <path-to-file> # für eine bestimmte Datei/Verzeichnis
```

```
git add -A # für alle Dateien
```

Änderungen commiten:

```
git commit -m "<kurze Beschreibung der Änderungen>"
```

Git Branching



Branches anzeigen:

```
git branch
```

Neue Branch anlegen:

```
git checkout -b <new-branch-name>
```

Branch am Remote anlegen:

```
git push --set-upstream origin <new-branch-name>
```

Branch wechseln:

```
git switch <branch-name>
```

`<branch-to-merge-from>` in aktuelle Branch mergen:

```
git merge <branch-to-merge-from>
```

Merge Konflikt

Wird dieselbe Datei gleichzeitig bearbeitet entsteht ein sogenannter Merge Konflikt und Git kann diese Änderungen nicht mehr selbständig mergen.

Beispiel: `file1.txt` mit folgendem Inhalt.

```
Hallo!
```

Wird nun in der `main` Branch wie folgt verändert:

```
Hallo, Welt!
```

Und in der `update-greeting` Branch:

```
Hallo, Alle!
```

Wird nun, nachdem alle Änderungen entsprechend committed sind, ein merge Versuch in die `main` Branch unternommen (`git merge update-greeting`) entsteht dabei ein Merge Konflikt. Und der Inhalt der Datei `file1.txt` ändert sich:

```
<<<<<<< HEAD
Hallo, Welt!
=======
Hallo, Alle!
```

```
>>>>>>> update-greeting
```

Wobei sich oben der Inhalt aus der aktullen Branch (`main`) und unten der Inhalt aus der Branch, aus welcher gemerged wurde (`update-greeting`), befindet. Nun muss der Konflikt manuell behoben werden und die Marker entfernt werden.

```
Hallo, gesamte Welt!
```

Nun können die Änderungen mit dem Befehl `git add` gestaged und der Mergevorgang mit `git commit` abgeschlossen werden.

.gitignore

Ist eine Datei, welche Regeln enthält, nach welchen Git Dateien ignoriert. Bei diesen Regeln handelt es sich um regular expressions.

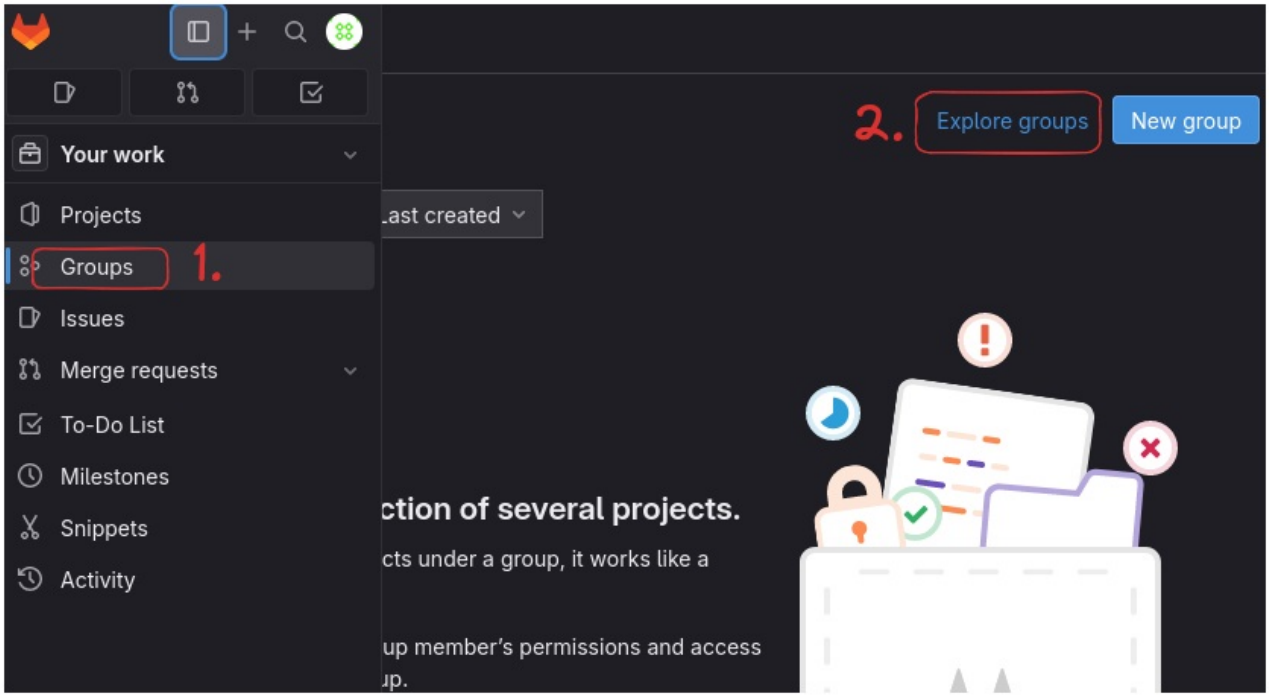
Zeichen	Beschreibung	Beispiel	Erklärung
`?`	ein beliebiges Zeichen	`?bar`	ignoriert `abar`, `bbar` aber nicht `foo` oder `foobar`
`*`	null oder mehrere Zeichen	`*.foo`	ignoriert alle Dateien, welche mit `.foo` enden
`/`	steht für Verzeichnisse	`/foo`	ignoriert die Datei `foo` nur im Projektstammverzeichnis
		`bar/`	ignoriert alle Verzeichnisse mit dem Namen `bar`
`**`	wie `*` nur auch über Verzeichnisgrenzen hinweg	`foo/**/bar`	ignoriert `foo/baz/bar` sowie `foo/baz/abc/bar`
`[]`	Gruppe von Zeichen	`[abc]foo`	ignoriert `afoo`, `bfoo` und `cfoo` aber nicht `wfoo`
		`[0-9]`	steht für die alle Zahlen von 0 bis 9
`!`	ignoriert diese Dateien nicht	`*.a` `!lib.a`	ignoriert alle Dateien, welche mit `.a` enden, außer `lib.a`

Sammlung von .gitignore templates für verschiedene Programmiersprachen.

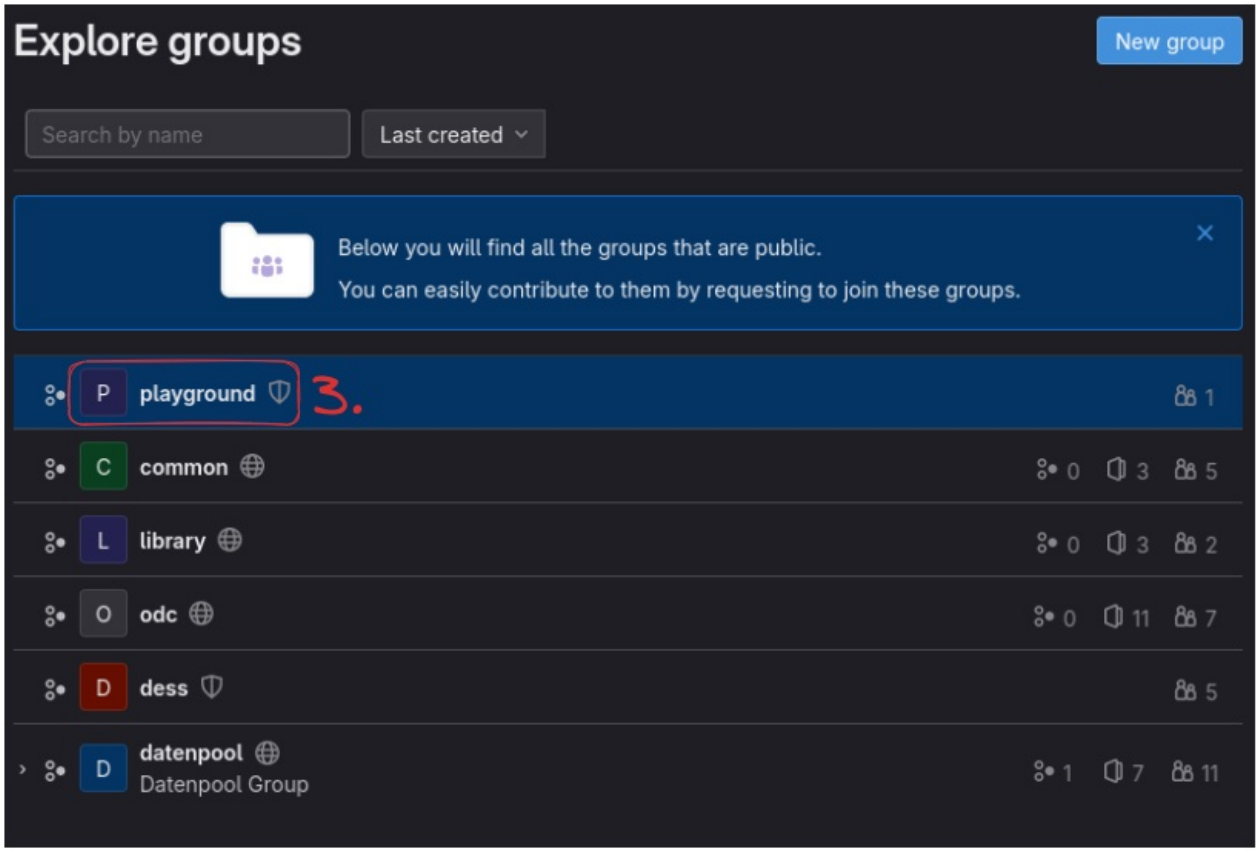
Gitlab

Einer Gruppe beitreten

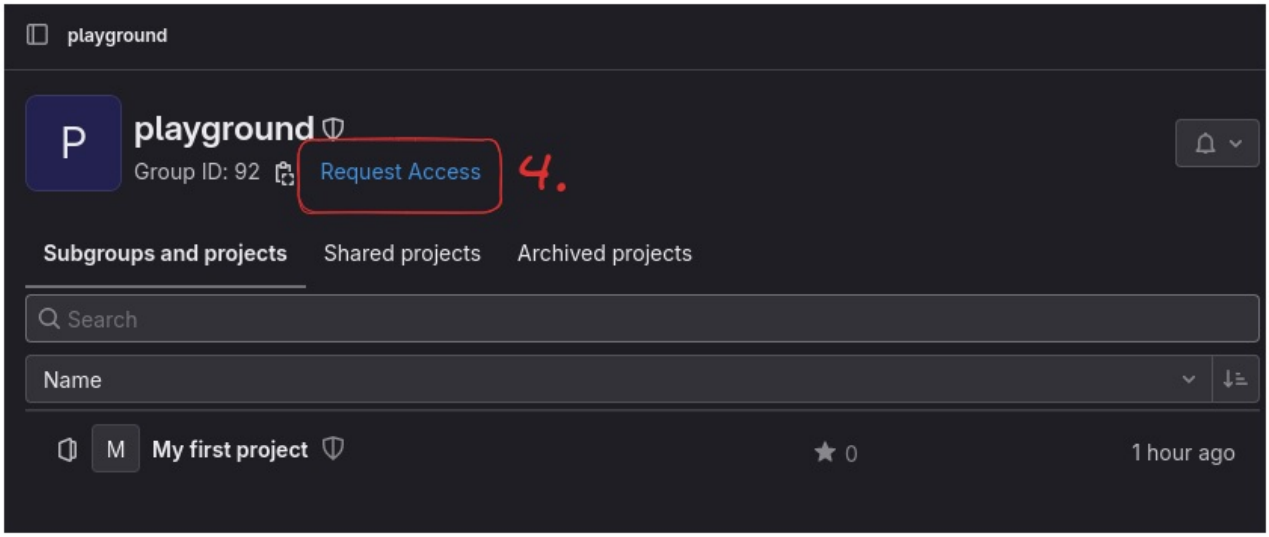
1. "Groups" auswählen
2. "Explore groups" auswählen



3. Gruppe auswählen

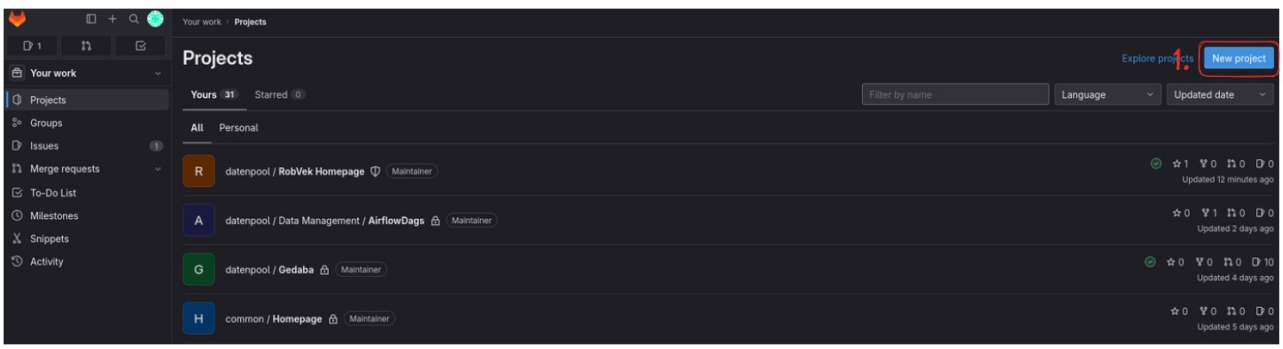


4. "Request Access" auswählen

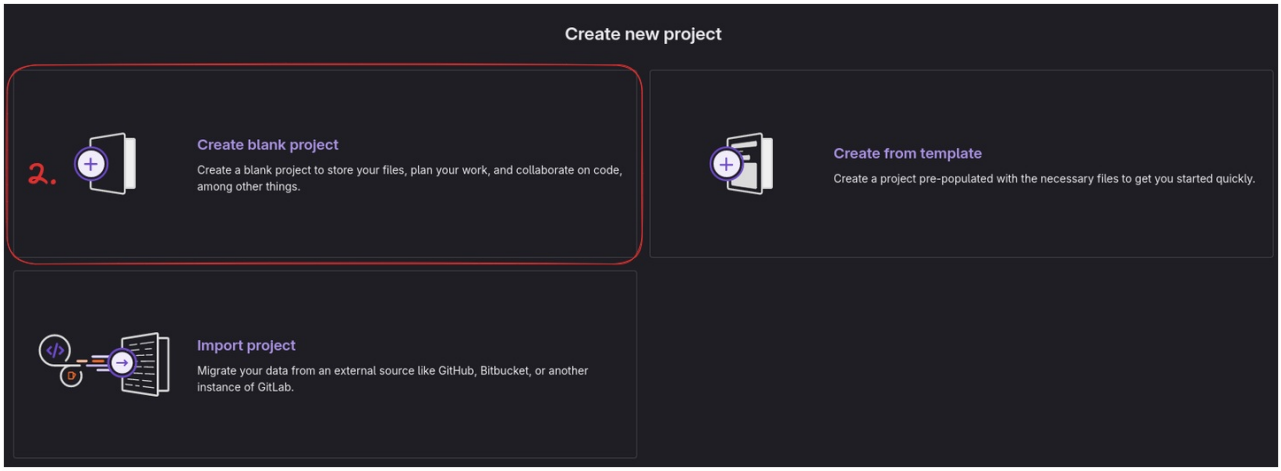


Neues Projekt anlegen

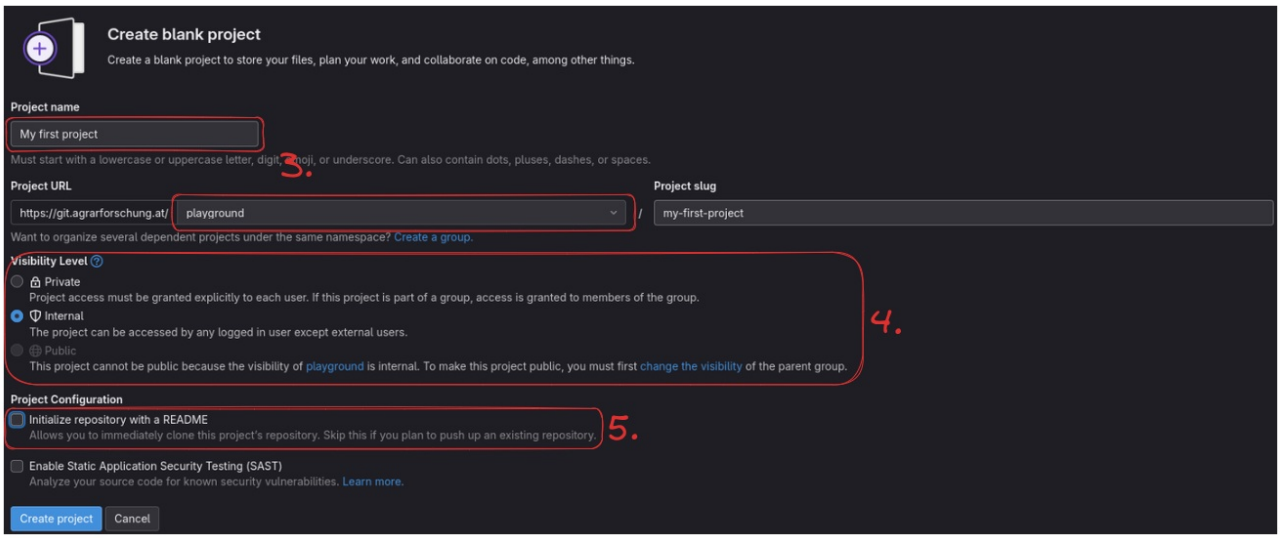
1. "New project" auswählen



2. "Create blank project" auswählen



3. Name und Gruppe festlegen
4. Visibility festlegen
5. Sollte bereits ein lokales Repository existieren, den Haken bei "Initialize repository with a README" entfernen



6. Anleitung auf der Website folgen

Command line instructions

You can also upload existing files from your computer using the instructions below.

Git global setup

```
git config --global user.name "Alex Leidwein"
git config --global user.email "alex.leidwein@bab.gv.at"
```

Create a new repository

```
git clone git@git.agrarforschung.at:playground/my-first-project.git
cd my-first-project
git switch --create main
touch README.md
git add README.md
git commit -m "add README"
git push --set-upstream origin main
```

Push an existing folder

```
cd existing_folder
git init --initial-branch=main
git remote add origin git@git.agrarforschung.at:playground/my-first-project.git
git add .
git commit -m "Initial commit"
git push --set-upstream origin main
```

Push an existing Git repository

```
cd existing_repo
git remote rename origin old-origin
git remote add origin git@git.agrarforschung.at:playground/my-first-project.git
git push --set-upstream origin --all
git push --set-upstream origin --tags
```

Weitere Ressourcen

- Offizielle Dokumentation
- `git help`
- `curl cht.sh/git`