

Migrationsworkshop GitLab

JOHANNES MICHAEL

Universität Rostock, IT- und Medienzentrum

Struktur des Workshops

Migration von **GitLab**

Direkt-Transfer

Export/Import-Funktionalität

Migration von **GitHub**

Alternativen

Aktualisieren lokaler Repositorien

Migration von GitLab



GitLab



GitLab

Migration mittels Direkt-Transfer

Direkter Import durch das Verbinden mit einer externen Instanz

- Es können mehrere Gruppen gleichzeitig migriert werden
- Gruppen enthalten alle Untergruppen
- Optional können Gruppen inklusive *all* ihrer Projekte importiert werden

Dokumentation: <https://docs.gitlab.com/ee/user/group/import/>

Direkt-Transfer: Einschränkungen

- Importieren von Gruppen inklusive ihrer Projekte ist noch ein "Beta-Feature"
- Importieren *persönlicher* Projekte ist *nicht direkt* möglich
- Auswählen *spezifischer* Projekte beim Migrieren von Gruppen ist nur über die [API](#) möglich.
- *Einzelne* Untergruppen können nur über die [API](#) migriert werden
- [Einige wenige Datenkategorien](#) werden bei der Migration **nicht** kopiert (z.B. CI/CD Variablen, Deploy Token, Container Registry, Package Registry)

Direkt-Transfer: Voraussetzungen

- Beide GitLab Instanzen idealerweise auf Version 16.8 (oder aktueller)
- Direkt-Transfer Feature **durch einen Admin aktiviert**
- Persönliches Zugriffstoken mit `api` Scope auf der ursprünglichen Instanz
- `owner` Rolle in der ursprünglichen Gruppe
- `owner` oder `Maintainer` Rolle im gewünschten Namespace der Ziel-Instanz

Direkt-Transfer: Mitgliedschaft & Nutzerbeiträge

- Initiale Zuordnung von importierten Mitgliedschaften und Nutzerbeiträgen zu Platzhalter-Nutzern
- Anschließende manuelle Zuordnung zu bestehenden Accounts
 - Nutzer:in kann Zuordnung annehmen oder ablehnen
 - Zuordnung kann zwischen beliebigen Accounts der ursprünglichen und Ziel-Instanz erfolgen
 - Zuweisungen können nicht rückgängig gemacht werden!
- Optional: permanente Platzhalter-Nutzer zuweisen
- Offene Mitgliedschaften bleiben als Platzhalter-Nutzer bestehen und können später zugewiesen werden

Direkt-Transfer: Persönliche Projekte

- Eigene Gruppe auf der ursprünglichen Instanz erstellen
 - *Create new (+ Symbol) > New group > Create group*
- Persönliche Projekte in die Gruppe **transferieren**:
 - Projekt öffnen
 - (Das Projekt darf keine Container Images enthalten)
 - *Project Settings > General > Advanced > Transfer project > Set group as new namespace*
- Gruppe inklusive Projekte mittels Direkt-Transfer migrieren

Direkt-Transfer: Workflow

1. Persönliches Zugriffstoken auf der ursprünglichen Instanz generieren:
 - *Avatar > Edit profile > Access Tokens > Add new token* mit Scope `api`
2. Auf GitLab *Create new (+ Symbol) > New group > Import group*
3. GitLab Instanz verbinden
 - Webadresse und persönliches Zugriffstoken
4. Gewünschten Namespace für die Gruppen setzen
 - Importieren als eigene Gruppe oder Untergruppe
5. Gruppe(n) importieren (mit oder ohne Projekte)
6. Bei Bedarf: Mitgliedschaften der Platzhalter-Nutzer den entsprechenden Nutzeraccounts zuweisen oder permanente Platzhalter-Nutzer setzen
 - *Group > Manage > Members > Placeholders*

Migration mittels Export/Import-Funktionalität

Datei-Export und anschließender Import in einer GitLab Instanz

- Kompatibel mit Gruppen und Projekten
- Exports werden über die Weboberfläche erzeugt
- Exports können über einen Email-Link oder über die Weboberfläche heruntergeladen werden
- Exports werden nach Erzeugen 24h aufbewahrt
- Imports werden über die Weboberfläche hochgeladen

Dokumentation: https://docs.gitlab.com/ee/user/project/settings/import_export.html

Export/Import: Einschränkungen

- Maximale Größe von Imports beschränkt auf 2 GB
 - Für größere Einzelfälle Support/Admin kontaktieren
- Gruppen und Projekte können nur **separat voneinander** ex-/importiert werden
- **Einige wenige Datenkategorien** werden bei Projekten **nicht** exportiert (z.B. CI/CD Variablen, Deploy Token, Container Registry, Package Registry)

Export/Import: Voraussetzungen

- Imports sind nur mit GitLab Exports kompatibel, die bis zu zwei minor Versionen zurückliegen (z.B. 17.1 $\leftarrow$ 17.1, 17.0, 16.10)
- `Owner` Rolle in der Gruppe zum Exportieren
- `Owner` oder `Maintainer` Rolle im Projekt zum Exportieren
- `Owner` oder `Maintainer` Rolle im Namespace zum Importieren
- Nutzerbeiträge und Zuordnung der Mitgliedschaft:
 - Accounts in der ursprünglichen Instanz müssen eine öffentliche Email haben
 - Accounts mit entsprechenden Emails müssen auf der Ziel-Instanz existieren
 - Hat ein Projekt durch eine Gruppe vererbte Mitglieder, muss ein `Owner` der Gruppe den Export veranlassen
 - *Import muss durch einen Admin erfolgen*, andernfalls werden keine Mitglieder zugewiesen

Export/Import: Workflow

Gruppe exportieren

1. Gruppe auf GitLab aufrufen
2. Gruppeneinstellungen wählen *Settings > General > Advanced > Export group*
3. Nachdem der Export erstellt wurde:
 - Den Link aufrufen, der in der Email zum Export enthalten ist, oder
 - Die Gruppeneinstellungen neu laden und unter *Export group* den Download auswählen

Export/Import: Workflow

Gruppe importieren

1. Auf GitLab *Create new* (+ Symbol) > *New group* > *Import group*
2. Im Abschnitt *Import group from file* Namen für die Gruppe vergeben
3. Gewünschten Namespace auswählen
4. Über *Choose file...* den Gruppen-Export auswählen
5. Import ausführen
6. Es kann sein, dass die Seite neu geladen werden muss, damit der Import angezeigt wird

Export/Import: Workflow

Projekt exportieren

1. Projekt auf GitLab aufrufen
2. Projekteinstellungen wählen *Settings* > *General* > *Advanced* > *Export project*
3. Nachdem der Export erstellt wurde:
 - Den Link aufrufen, der in der Email zum Export enthalten ist, oder
 - Die Projekteinstellungen neu laden und unter *Export project* den Download auswählen

Export/Import: Workflow

Projekt importieren

1. Auf GitLab *Create new (+ Symbol) > New project/repository > Import project > GitLab export*
2. Name für das Projekt vergeben
3. Gewünschten Namespace auswählen
4. Über *Choose file...* den Projekt-Export auswählen
5. Import ausführen
6. Es kann sein, dass die Seite neu geladen werden muss, damit der Import angezeigt wird

Migration von GitHub

GitHub →  **GitLab**

Migration von GitHub

Direktes Importieren von Projekten von github.com

- Importieren von inoffiziellen GitHub Instanzen ist nur über die [GitLab API](#) möglich
- Es können keine Gruppen oder Organisationen importiert werden
- Import enthält Git-Repository, Issues, Pull Requests, Kommentare, Reviews, (Mitglieder) etc. ([siehe vollständige Liste](#))
- GitHub Actions (CI/CD) können nicht automatisch konvertiert werden

Dokumentation: <https://docs.gitlab.com/ee/user/project/import/github.html>

Migration von GitHub: Voraussetzungen

- Mindestens die `Maintainer` Rolle im GitLab Namespace
- Authentifikation nutzt *klassisches* persönliches Zugriffstoken mit `repo` Scope
 - Zur Migration von Mitgliedern oder LFS Dateien, zusätzlich `read:org` Scope
- Nutzerbeiträge und Zuordnung der Mitgliedschaft:
 - Mitglieder in GitHub müssen eine öffentliche Email-Adresse haben. Ein Account mit der entsprechenden Email muss auf GitLab existieren.
 - Andernfalls werden keine Mitglieder zugewiesen und der Importeur als Autor gesetzt

Migration von GitHub: Workflow

1. *Klassisches* persönliches Zugriffstoken auf GitHub generieren:
 - *Settings > Developer Settings > Personal access tokens > Tokens (classic) > Generate new token (classic)*
 - Oder folgende Adresse besuchen: <https://github.com/settings/tokens/new>
 - Scope `repo` auswählen (und optional `read:org` für Mitglieder und LFS Dateien)
2. Auf GitLab *Create new (+ Symbol) > New Project/repository > Import Project > GitHub*
3. Persönliches Zugriffstoken von GitHub einfügen
4. Projekte importieren:
 - Projekt aus Untermenüs auswählen
 - Namespace in GitLab anpassen
 - Optionale Daten auswählen

Alternativen



Alternativen

Folgende Importquellen sind derzeit **deaktiviert**

- Bitbucket Cloud/Server
- FogBugz
- Gitea
- Manifest File

Als Alternativlösung kann der Import "[Repository via URL](#)" verwendet werden.

- Angabe einer konkreten Git-URL
- Optionale Authentifikation für nicht-öffentliche Projekte
- Import enthält ausschließlich das Git Repository

Was, wenn nicht Git?

Migrationen anderer VCS werden durch GitLab **nicht direkt unterstützt**, aber es gibt externe Tools für die Konvertierung:

System	Weiterführende Informationen
Subversion (SVN)	Link
ClearCase	Link
Concurrent Versions System (CVS)	Link
Perforce Helix	Link
Team Foundation Version Control (TFVC)	Link

Aktualisieren lokaler Repositorien

Nach der Migration von Projekten, verweisen die Remote-Pfade in bestehenden lokalen Repositorien immer noch auf die ursprünglichen Instanzen und müssen aktualisiert werden.

Falls man keine lokale Kopie des alten Projektes hat oder dieses verwerfen möchte, sind diese Schritte nicht notwendig. Man kann einfach das neue Projekt wie gewohnt klonen.

Aktualisieren lokaler Repositorien: Voraussetzungen

- Ein GitLab Projekt mit Git Repository und neuer Git-URL
- Eine lokale Kopie des alten Repository
- Authentifizierungsdaten für GitLab
 - Für HTTPS: Persönliches Zugriffstoken
 - Für SSH: Hinterlegter SSH-Schlüssel

Aktualisieren lokaler Repositorien: Workflow

Remote-Pfade und Git-URLs bestimmen

1. Im Terminal zur lokalen Kopie navigieren:

```
cd <repository_name>
```

2. Anzeigen der aktuellen Remote-Pfade für *fetch* und *push* URLs (diese stimmen normalerweise überein und heißen *origin*):

```
git remote -v
```

3. Neue Git-URL aus GitLab holen

- Projekt auf GitLab aufrufen
- Blaue Schaltfläche *Code* auswählen
- Abhängig von der gewünschten Authentifikationsmethode entweder die HTTPS oder SSH URL kopieren

Aktualisieren lokaler Repositorien: Workflow

Git-Remote-URL aktualisieren

1. Remote-URL mit der zuvor kopierten URL überschreiben:

```
git remote set-url origin <new_url>
```

2. Um ein anderes Repository als *origin* zu aktualisieren, kann *origin* im obigen Befehl durch den jeweiligen Namen ersetzt werden.

3. Neue Remote-URL verifizieren:

```
git remote show origin
```

4. Ab jetzt sollten alle Git Befehle wie gewohnt funktionieren und mit dem neuen Repository interagieren.

Vielen Dank für Ihre Aufmerksamkeit!
Gibt es **Fragen**?

GitLab Support
gitlab@uni-rostock.de