Git Repository für M183

Diese Kurzanleitung können Sie für die Einrichtung eines Git Repositories auf GitLab für die Verwendung im Modul 183 am GIBZ benutzen.

Repository anlegen

Auf https://gitlab.com müssen Sie sich registrieren, sofern Sie noch keinen Account besitzen.

Nach der Anmeldung können Sie in der Menüleiste, links neben dem Suchfeld auf as `+` Icon und anschliessend auf `New project` klicken, um ein neues Projekt (inkl. Git Repository) anzulegen.

Als Projektart wählen Sie ein neues, leeres Projekt (`Create blank project`).

Geben Sie dem Projekt einen geeigneten Namen und setzen Sie die Sichtbarkeit (`Visibility level`) auf privat (`Private`). Je nach Bedarf können Sie die Option zum automatischen Erstellen einer README-Datei auswählen (`Initialize repository with a README`).

Mit einem Klick auf `Create project` können Sie das Projekt anschliessend erstellen.

SSH Schlüssel

Am einfachsten können Sie von Ihrem Gerät mit dem Repository, welches auf GitLab gehostet ist, über SSH interagieren. Dazu müssen Sie ein Schlüsselpaar generieren und den öffentlichen Schlüssel auf GitLab hinterlegen.

SSH Schlüsselpaar generieren

Starten Sie auf Ihrem Gerät ein Kommandozeilenprogramm (Terminal, PowerShell, Eingabeaufforderung CMD, ...) und geben Sie den nachfolgenden Befehl ein, um ein RSA-Schlüsselpaar zu generieren:

```
ssh-keygen -t rsa -b 2048 -C "your-email-address@domain.ch"
```

Den Wert, welchen Sie nach dem `-C` Parameter angeben, können Sie für die Bezeichnung des Schlüsselpaares verwenden. Üblicherweise wird dieser Parameter für die Angabe einer E-Mail Adresse, eines Benutzernamens oder einer sonstigen Bezeichnung verwendet.

Das Tool `ssh-keygen` führt Sie anschliessend durch die Erstellung eines Schlüsselpaares. Die meisten vorgeschlagenen Optionen können Sie ohne Eingabe eines weiteren Wertes akzeptieren:

- Den vorgeschlagenen Pfad für den Speicherort sollte typischerweise unverändert übernommen werden. Sie können `ENTER` drücken, ohne einen Pfad einzugeben, um den vorgeschlagenen Wert in der Klammer zu akzeptieren.
- Die Eingabe einer Passphrase ist möglich - jedoch nicht zwingend erforderlich. Um ein Schlüsselpaar *ohne* Passphrase zu erzeugen, drücken Sie bei der entsprechenden Aufforderung (sowie bei der Bestätigung) `ENTER` ohne Eingabe einer Passphrase.

Nach Abschluss sind im angegebenen Verzeichnis die zwei Bestandteile des SSH Schlüsselpaares erzeugt worden - ein öffentlicher Teil mit der Endung `.pub` sowie ein privater Teil mit dem identischen Dateinamen, jedoch ohne erkennbare Dateiendung.

SSH Schlüssel zum Konto auf GitLab hinzufügen

Damit Sie künftig eine SSH Verbindung zwischen Ihrem Gerät und dem GitLab Server herstellen können, müssen Sie als nächstes den öffentlichen Teil des Schlüsselpaares auf GitLab hinterlegen. Dazu müssen Sie den Inhalt der soeben erstellten Datei mit der `.pub` Endung kopieren und anschliessend auf GitLab hinterlegen.

Um den Inhalt in die Zwischenablage zu kopieren, können Sie - abhängig von Ihrem Betriebssystem - den nachfolgenden Befehl nutzen. Alternativ können Sie die jeweilige Datei auch in einem Texteditor öffnen und den Inhalt manuell kopieren.

Windows (Git Bash):

```
cat ~/.ssh/id_rsa.pub | clip
```

macOS:

```
pbcopy < ~/.ssh/id_rsa.pub
```

Anschliessen müssen Sie sich auf GitLab anmelden und in der oberen, rechten Ecke auf Ihr Profilbild klicken. Wählen Sie anschliessend den Menüpunkt *Settings* aus. In der linken Seitennavigation finden Sie etwas unterhalb der Mitte den Eintrag *SSH Keys* - klicken Sie diesen an. Fügen Sie anschliessend den Inhalt in das Eingabefeld mit dem Titel *Key* ein. Ein Ablaufdatum ist nicht erforderlich - Sie können den Schlüssel jederzeit wieder manuell entfernen. Klicken Sie zum Abschliessen des Vorgangs auf den Button *Add key*.

SSH Verbindung testen

Sie können die SSH Verbindung testen, indem Sie in der Eingabeaufforderung den nachfolgenden Befehl eingeben:

```
ssh -T git@gitlab.com
```

Wenn Sie zum ersten Mal eine SSH Verbindung mit `gitlab.com` aufbauen, müssen Sie durch die Eingabe von `yes` den *Fingerprint* des Servers zur Liste Ihrer bekannten Server ("known hosts") hinzufügen:

```
The authenticity of host 'gitlab.com (35.231.145.151)' can't be
established.
ECDSA key fingerprint is
SHA256:HbW3g8zUjNSksFbqTiUWPWg2Bq1x8xdGUrliXFzSnUw.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'gitlab.com' (ECDSA) to the list of known hosts.
```

Wenn mit der Verbindung alles klappt, sollten Sie in der Ausgabe der Kommandozeile eine entsprechende Begrüssung lesen können:

```
Welcome to GitLab, @USERNAME!
```

Weitere Unterstützung

Sie finden eine detaillierte Anleitung mit weiteren Informationen auf der entsprechenden Seite von GitLab: https://gitlab.com/help/ssh/README#generating-a-new-ssh-key-pair

Repository klonen

Um den Inhalt des (vermutlich noch ziemlich leeren) Repositories auf Ihr Gerät zu kopieren, müssen Sie dieses im nächsten Schritt klonen. Verwenden Sie für diesen Vorgang die URL, welche Sie bei Ihrem Projekt mit dem blauen `Clone` Button unter `Clone with SSH` kopieren können.

Navigieren Sie mit Ihrer Kommandozeilen-Anwendung an den Ort auf Ihrem Gerät, an welchem Sie das Repository klonen möchten. Durch das Klonen wird Git ein neues Verzeichnis mit dem Namen Ihres Repositories anlegen. Der Inhalt des Repositories wird anschliessend in diesem neu erstellten Verzeichnis sein.

```
git clone git@gitlab.com:USERNAME/m183.git
```

Inhalte hinzufügen

Sie können anschliessend mit einem beliebigen Programm Inhalte zu Ihrem Repository hinzufügen, indem Sie innerhalb des soeben geklonten Repositories neue Dateien anlegen. Im Modul 183 geschieht dies in der Regel dadurch, dass Sie für jede Praxisaufgabe ein neues Verzeichnis anlegen (z.B. `01 Keylogger`) und anschliessend innerhalb dieses Verzeichnisses neue Dateien anlegen.

Alternativ erhalten Sie für verschiedene Praxisaufgaben eine Projektvorlage. Diese Projektvorlage können Sie hinzufügen, indem Sie die entsprechenden Dateien zu einem Verzeichnis innerhalb Ihres Repetitories hinzufügen (z.B. Verzeichnis `02 Zweifaktorauthentifizierung` anlegen und anschliessend Projektovorlage in dieses neue Verzeichnis kopieren).

Staging

Mit dem alleinigen Hinzufügen von neuen Dateien zu einem Verzeichnis, werden diese neuen Dateien nicht automatisch durch Git verwaltet. Um die Versionsverwaltung für die neuen Datein zu starten, muss Git angewiesen werden, diese Dateien zu überwachen. In Git heisst dieser Prozess `staging` und geschieht mit dem Befehl `add`.

Um Beispielsweise alle Dateien im Verzeichnis `02_Zweifaktorauthentifizierung` zur Versionsverwaltung hinzuzufügen, können Sie in der Kommandozeilen-Anwendung folgenden Befehl verwenden:

```
1 git add 02_Zweifaktorauthentifizierung/
```

Ob der Vorgang erfolgreich war, können Sie anschliessend mit dem Befehl `status` überprüfen. Die Eingabe des nachfolgenden Befehls listet Ihnen alle Dateien auf, welche durch den vorangehenden `add` Befehl neu zur Versionsverwaltung hinzugefügt wurden:

```
1 git status
```

Commit

Mit einem `commit` werden alle Änderungen an den Dateien, welche durch Git überwacht werden (also jene Dateien, welche zuvor mit dem Befehl `add` zur Versionskontrolle hinzugefügt wurden), in die *lokale Versionskontrolle* übernommen.

Wichtig: Mit einem `commit` werden die Änderungen noch *nicht* an den GitLab Server übertragen!

Navigieren Sie in der Kommandozeilen-Anwendung auf die Wurzelebene Ihres Repositories (dort wo sich das `.git` Verzeichnis befindet) und führen Sie den nachfolgenden Befehl aus:

```
1 git commit -m "commit message"
```

Der Parameter `-m` wird für die Angabe einer *Commit Message* verwendet. Beschreiben Sie in dieser Message die relevanten Änderungen am Code, welche seit dem letzten `commit` erfolgt sind.

Der `commit` war erfolgreich, wenn der Befehl `git status` den Hinweis enthält, dass keine Dateien für ein `commit` vorhanden sind: `nothing to commit, working tree clean`.

Push

Um die Änderungen der *lokalen Versionskontrolle* auf die Server von GitLab zu übertragen (in die *zentrale Versionskontrolle*), müssen diese Änderungen mit dem Befehl `push` aktiv übertragen werden.

Führen Sie dazu den folgenden Befehl auf der Kommandozeile aus:

```
1 git push
```

Dadurch werden die Änderungen des aktuellen Arbeitszweiges (*branch*) auf einen gleichnamigen Arbeitszweig der *zentralen Versionskontrolle* übertragen.

Sie können diesen Vorgang überprüfen, indem Sie sich auf GitLab anmelden und den Inhalt Ihres Repositories auf der Weboberfläche überprüfen.