Selbststudium Git

1 Einführung

In diesem Dokument lernen Sie die Grundlagen von Git und Github kennen. Diese benötigen Sie für das Vorpraktikum.

1.1 Versionskontrolle

Software zur Versionskontrolle (Version Control System - VCS) hilft Software Entwicklern und Teams Änderungen am Source Code zu verwalten. Dabei werden Änderungen in einer Art Datenbank gespeichert und können somit nachverfolgt werden. So können zum Beispiel Änderungen einfach rückgängig gemacht werden.

1.2 Git

Git ist das am meisten benutzte Tool, wenn es um Versionskontrolle geht. Git ist Open Source und kann mittels Terminal bedient werden. Zusätzlich gibt es verschiedene grafische Tools sowie Plugins für Entwicklungsumgebungen.

1.3 Github

Github ist eine Platform zur Verwaltung von Git Projekten. Aufbauend auf Git bietet Github weitere nützliche Funktionen. Dies sind zum Beispiel ein Issue Tracker, um Tasks, Bugs oder andere Probleme als Tickets zu verwalten. Github gibt es als öffentliche Platform (github.com), die jeder frei verwenden kann. An der ZHAW gibt es ein Enterprise Github (github.zhaw.ch), das für Mitarbeiter und Studenten zur Verfügung steht.

2 Grundlagen

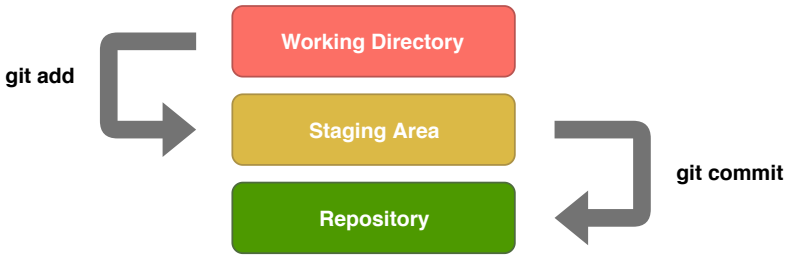
2.1 Repository

Mit Git wird der Source Code von einer Applikation oder einer Software-Bibliothek in einem **Repository** verwaltet. Ein Repository besteht aus aufeinander aufbauenden Commits. Ein **Commit** ist eine Menge von Änderungen am Source Code. Jeder Commit ist eindeutig über eine ID identifizierbar und hat eine Beschreibung. Die Beschreibung ist wichtig und soll klar zu erkennen geben, was für Änderungen in diesem Commit gemacht wurden. Im folgenden Bild sehen Sie ein Beispiel einer **Commit-History**.

```
2e10426: Added a line of text to my_file.txt.
91f1fef: Converted another_file.txt to a Bash script.
1ddccf6: Added a new file and added several lines to my_files.txt.
2f06323: First commit. Added an important file.
```

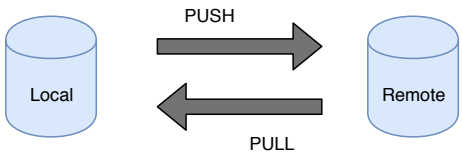
2.2 Workflow

Um einen Commit in einem Repository zu erstellen werden verschiedene Phasen durchlaufen. Als **Working Directory** oder **Working Tree** bezeichnet man die Dateien und Änderungen, an welchen Sie aktuell arbeiten und noch nicht durch Git verwaltet sind. Sobald Sie etwas zu Git hinzufügen möchten werden die gewünschten Änderungen in die **Staging Area** hinzugefügt. Daraus wird dann ein Commit erstellt und somit dem **lokalen Repository** hinzugefügt.



2.3 Remote

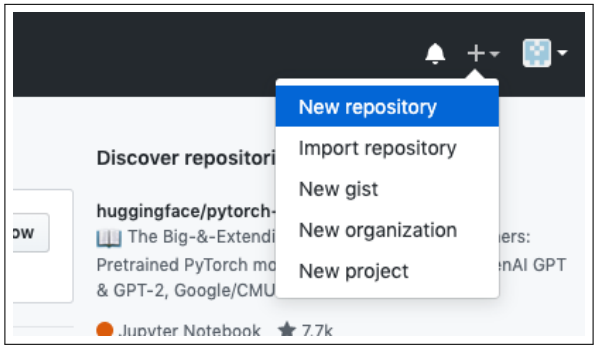
Mit Git haben Sie lokal ein Repository, mit welchem Sie arbeiten. Nun ist es aber notwendig die Änderungen für andere zur Verfügung zu stellen oder einfach nur zur Sicherheit an einem anderen Ort zu speichern. Dafür gibt es ein **Remote Repository**. Dieses wird zum Beispiel durch Github verwaltet. Das Remote Repository ist ein Abbild des Repositorys, welches Sie lokal bei Ihnen auf dem PC haben. Git stellt die Befehle zur Verfügung, um Commits auf ein Remote Repository zu übertragen (push) oder von einem Remote Repository herunterzuladen (pull).



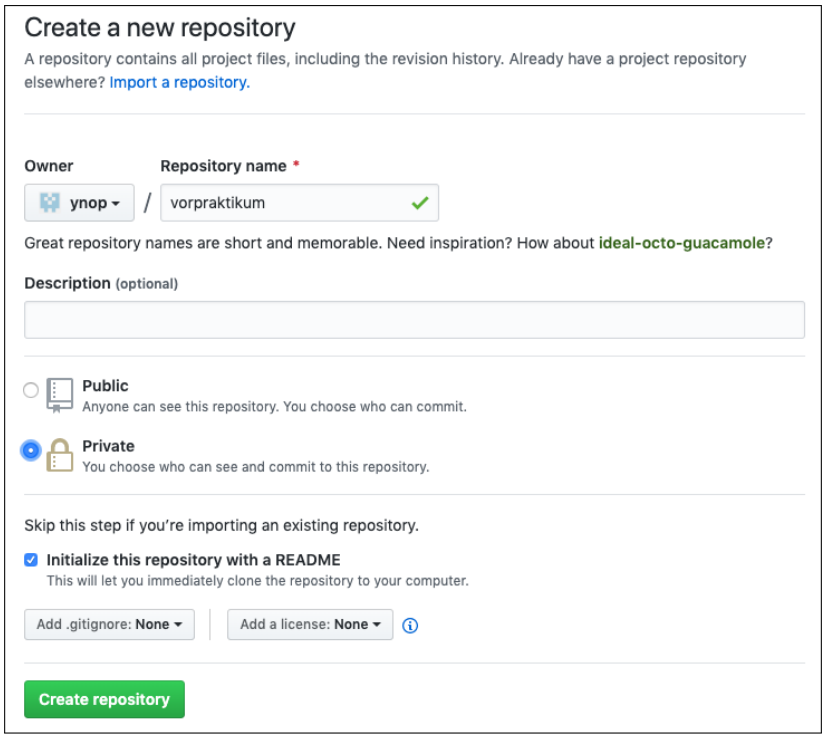
3 Repository erstellen

Wenn Sie ein neues Projekt starten, müssen Sie in einem ersten Schritt ein neues Repository für Ihr Projekt erstellen. Dies machen wir über die Webseite von Github.

In einem ersten Schritt erstellen Sie einen Account auf github.com, falls Sie nicht bereits einen besitzen. Erstellen Sie dann ein neues Repository.



Verwenden Sie folgende Einstellungen. Diese sind vorerst nicht wichtig, so haben Sie aber diesselbe Ausgangslage.



4 Repository verwenden

Nachdem Sie ein Repository erstellt haben, wollen Sie ja auch Ihren Source Code darin verwalten. Dafür wird das Repository lokal geklont. Das heisst Sie haben dann ein Verzeichnis auf Ihrem PC welches den Inhalt des Repositorys hat. Um das Repository zu klonen und Änderungen zu verwalten benötigen Sie ein Tool. Es gibt

verschiedene Möglichkeiten. Im folgenden sind zwei Varianten beschrieben, einmal mit einem grafischen Tool und einmal über die Kommandozeile. Zuerst wird allerdings beschrieben, wie man mit dem Repository über die Github Webseite arbeitet. Diese bietet allerdings nur eingeschränkte Möglichkeiten und ist nicht geeignet um Source Code zu bearbeiten.

4.1 Github Webseite

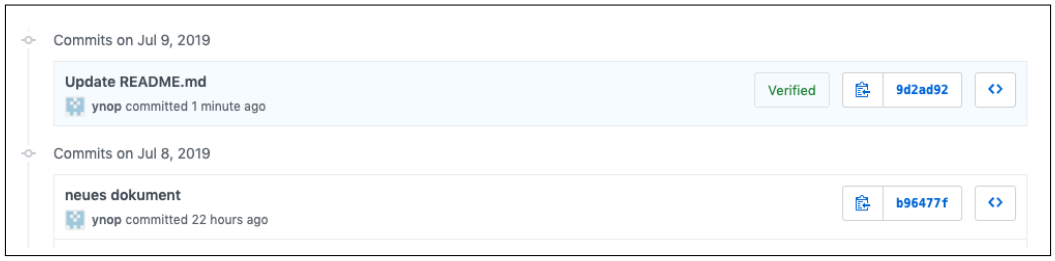
4.1.1 Files erstellen/bearbeiten

Erstellen Sie zuerst ein neues File mit dem Button 'Create new file' auf der Hauptseite des Repositorys. Es öffnet sich ein Editor, wo Sie den Inhalt des Files definieren können. Fügen Sie einen beliebigen Text ein. Speichern Sie das File, indem Sie eine Beschreibung hinzufügen und auf 'Commit new file' klicken.

Auf dieselbe Art können Sie auch bestehende Files bearbeiten. Klicken Sie auf das eben erstellte File und ergänzen Sie es um einen beliebigen Text. Den Editor können Sie über das Bearbeitungs-Icon öffnen. Erneut speichern Sie Ihre Änderungen, indem Sie auf 'Commit changes' klicken.

4.1.2 Historie anzeigen

Um die Historie der Änderungen anzuzeigen klicken Sie auf der Hauptseite des Repository auf 'X commits'. Sie sehen alle Änderungen, die Sie bereits gemacht haben.



Klicken Sie auf eine der Änderungen. Sie sehen nun eine detailierte Aufstellung der Änderungen, die Sie in diesem Commit gemacht haben.

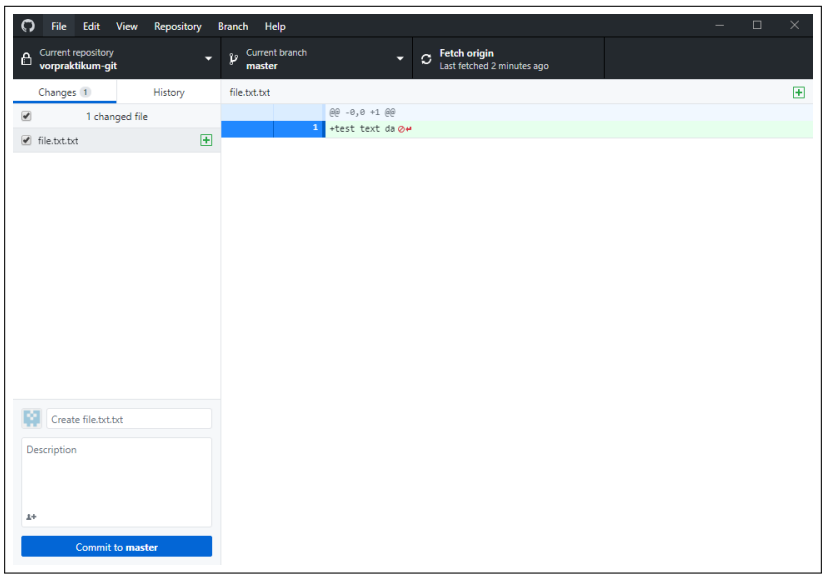
4.1.3 Issue anlegen

Eine weitere nützliche Funktion, die Github anbietet sind Issues. Die dienen dazu um zum Beispiel Bugs oder zu erledigende Aufgaben zu erfassen. Wechseln Sie zu den Issues über den Reiter 'Issues'. Erstellen Sie ein neues Issue. Setzen Sie einen beliebigen Titel und eine Beschreibung. Zusätzlich können Sie Labels definieren, um das Issue einer bestimmten Kategorie zuzuweisen.

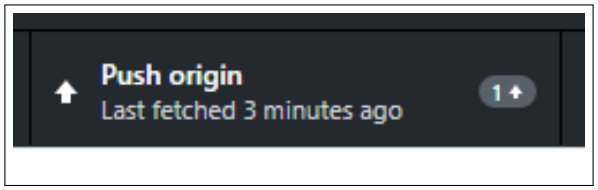
4.2 Github Desktop

Eine Möglichkeit, um mit einem Git Repository zu arbeiten ist der Github Desktop Client. Installieren Sie diesen von desktop.github.com. Starten Sie die Applikation und loggen Sie sich ein. Der Name und die Email im Konfigurations-Schritt werden verwendet, um Änderungen am Repository mit Ihrem User Account zu verknüpfen. Wenn die Applikation gestartet ist, sehen Sie eine Liste ihrer Repositories. Klonen Sie nun das Repository, das Sie zuvor erstellt haben. Dadurch wird ein lokales Repository vom Remote Repository erstellt (Standardmässig unter *Documents \GitHub*).

Öffnen Sie nun das Repository im File Explorer und erstellen Sie eine neue Text-Datei und schreiben Sie etwas darin. Dies wäre nun der Source Code, den Sie commiten wollen. Gehen Sie wieder zu Github Desktop zurück. Sie sehen nun die Änderungen, die Sie gemacht haben (Working Directory). Durch das Häckchen vor dem Datei-Namen können Sie die Datei in die Staging Area hinzufügen oder entfernen.



Fügen Sie eine Beschreibung Ihrer Änderungen hinzu und klicken Sie auf 'Commit to master'. Der Commit ist nun auf ihrem lokalen Repository erstellt. Um die Änderungen auf das Remote Repository hochzuladen klicken Sie auf 'Push origin'.



Um Änderungen vom Remote Repository herunterzuladen verwenden Sie 'Fetch origin' und danach 'Pull origin'. Testen Sie das, indem Sie ein neues File über die Webseite von Github erstellen.

4.3 Terminal

Eine andere Möglichkeit, um mit Git zu arbeiten ist die Kommandozeile. Hier verwenden Sie textuelle Befehle, um die verschiedenen Funktionen zu nutzen. Installieren Sie dazu Git von git-scm.com. Um die Kommandozeile zu öffnen, suchen Sie nach dem Programm 'Command Prompt'.

4.3.1 Repository klonen

Als erstes müssen Sie ein lokales Abbild des Remote Repository erstellen. Wechseln Sie in das Verzeichnis, wo Sie die lokale Kopie erstellen möchten.

```
cd verzeichnis
```

Verwenden Sie nun den 'clone' Befehl. Ersetzen Sie dabei die URL mit der Ihres Repositorys.

```
git clone https://github.com/ynop/vorpraktikum-git.git
```

4.3.2 Änderungen hinzufügen

Erstellen Sie im Repository ein neues File mit beliebigem Inhalt. Mit dem 'status' Befehl, können Sie nun sehen welche Änderungen es im Repository gibt.

```
git status
```

Im Bereich 'Untracked files' sehen Sie die Änderungen im Working Directory. Verwenden Sie den 'add' Befehl, um die Änderungen in die Staging Area hinzuzufügen.

```
git add .
```

Sie können auch nur einzelne Files hinzufügen, indem Sie den genauen Pfad angeben. Die Änderungen in der Staging Area erscheinen nun unter 'Changes to be committed'. Nun können Sie einen Commit im lokalen Repository erstellen.

```
git commit -m "beschreibung"
```

Nun pushen Sie den Commit noch auf das Remote Repository.

```
git push
```

4.3.3 Änderungen abrufen

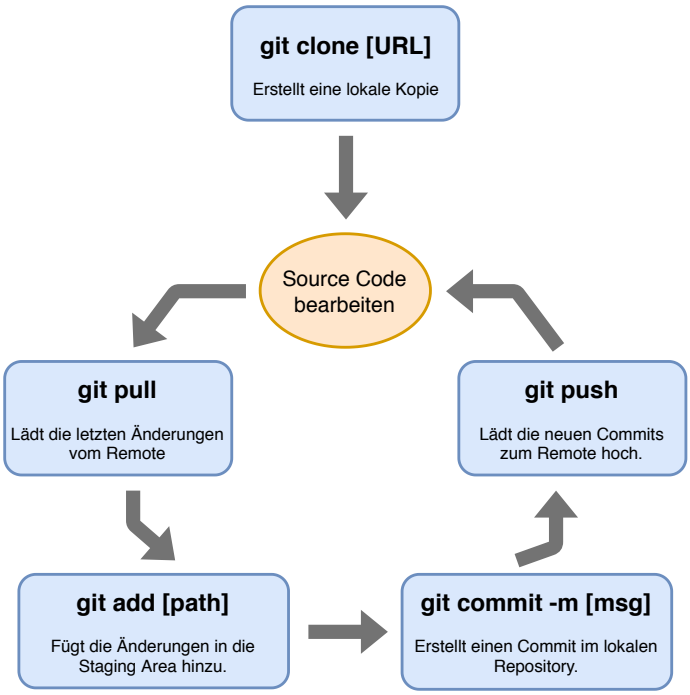
Um neue Commits vom Remote Repository herunterzuladen verwenden Sie den 'pull' Befehl.

```
git pull
```

Um das auszuprobieren erstellen Sie zuerst einen neuen Commit über die Github Webseite oder den Github Desktop Client.

4.3.4 Best Practice

Arbeiten mehrere Personen mit demselben Repository kann es zu Konflikten kommen, wenn gleichzeitig an einem File gearbeitet wird. Daher ist es sinnvoll einen gewissen Ablauf einzuhalten, um Konflikte bestmöglich zu vermeiden, auch wenn das nicht immer möglich ist.



5 Weiterführende Informationen

Verwenden Sie die verschiedenen Befehle und Funktionen, die bisher beschrieben wurden. Simulieren Sie zwei Entwickler indem Sie ein Repository einmal mit dem Desktop Client und einmal mit dem Terminal klonen. Sie können auch dasselbe Repository zweimal via Terminal an unterschiedlichen Orten klonen.

Im Internet gibt es zudem eine Menge an Ressourcen zum Thema Git und Github. Bei Unklarheiten oder wenn Sie genauere Information zu einem Befehl brauchen machen Sie davon Gebrauch. Einige hilfreiche Ressourcen sind zum Beispiel:

https://git-scm.com/book/en/v2 Ein komplettes Buch mit Ausführungen zu Versionskontrolle und Git. Ausserdem eine gute Beschreibung aller Git Befehle.

https://www.atlassian.com/git/tutorials Eine Reihe von Tutorials zum lernen der Git Befehle.

https://learngitbranching.js.org/ Eine Möglichkeit, um Git interaktiv zu erlernen und zu üben. Befehle werden beschrieben und können direkt ausprobiert werden.

https://www.youtube.com/ Es gibt viele gute Videos, die Git und dessen Funktionen beschreiben.