Leitfaden Git

Hochschule für Angewandte Wissenschaften (HAW), Hamburg
Autor: Prof. Dr. Philipp Jenke
Stand: 22.04.2015

Hinweis: Dieser Leitfaden soll als einfache Referenz für den Einstieg in Git dienen. Die Funktionalität wird beispielhaft und nicht vollständig beschrieben.

Software
Git ist für alle gängigen Betriebssysteme verfügbar. Git ist ein Kommandozeilen-Werkzeug, es werden aber auch grafische Benutzeroberflächen angeboten. Dieser Leitfaden beschränkt sich auf die Bedienung per Kommandozeile. Installer dafür können hier heruntergeladen werden: *http://git-scm.com/*. Dort finden Sie auch Informationen dazu, wie Sie mit der Kommandozeilen-basierten Eingabe von Git-Befehlen starten.

Konfiguration
Zunächst wird Git mit Ihren Anwenderdaten konfiguriert (Name und Mailadresse).
```
git config --global user.name "< Name>"
git config --global user.email "<E-Mail>"
```

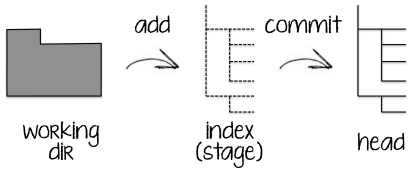

Abbildung 1: Struktur des lokalen Repositories.

Anlegen eines Repositories
Zum Anlegen eines neuen Repositories wechseln Sie in das Verzeichnis, in dem das Repository liegen soll.
Beispiel:
```
mkdir pm1
cd pm1
```
In dem Verzeichnis wird das Repository mit dem Kommando `init` erzeugt:
```
git init
```
In dem Verzeichnis sollte sich nun ein (verstecktes) Verzeichnis `.git/` befinden.

Anlegen eines Repositories
Möchte man mit einem Repository weiterarbeiten, das bereits an einer anderen Stelle (z.B. auf einem anderen Rechner) existiert, dann verwendet man das Kommando `clone`:
```
git clone <url>
```
Beispiel (Repo-Verzeichnis `/git/pm1` auf Rechner unter `git.server.de`):
```
git clone git.server.de:/git/pm1
```

Einfügen von Dateien
Die Dateien, die Sie mit dem Repository verwalten wollen, werden ebenfalls in diesem Verzeichnis abgelegt. Beispielsweise könnten Sie dort ein neues Eclipse-Projekt erzeugen. Dessen Quellcode könnte z.B. im Unterverzeichnis `src` liegen. Um Dateien aus dem Dateisystem in das Repository einzufügen, müssen sie mit dem Kommando `add` hinzugefügt werden.
```
git add <Datei-oder-Verzeichnis>
```
Beispiel (Einfügen der Datei `src/HalloWelt.java`):
```
git add src/HalloWelt.java
```

Änderungen in das Repository einpflegen
Änderungen an den Dateien eines Repositories werden mit dem Kommando `commit` in das Repository übertragen. Dies ist auch notwendig nachdem neue Dateien mit `add` hinzugefügt wurden:
```
git commit -m "<Beschreibung>"
```
Um alle Änderungen in das Repository zu übernehmen, muss der Aufruf im Wurzelverzeichnis des Projektes ausgeführt werden (im Beispiel in `pm1`). Alternativ ist es auch möglich, `commit` nur in einem Unterverzeichnis zu anzuwenden.

Status des Repositories anzeigen
Den aktuellen Zustand des lokalen Repositories, insbesondere die Unterschiede zwischen *index* und *head* (siehe Abb. 1) kann man mit dem Kommando `status` anzeigen lassen:
```
git status
```

Verbinden mit anderem Repository
Ein Git-Repository kann mit anderen Repositories auf anderen Rechnern verbunden werden. Dazu legt man zunächst eine Verbindung zu einem "remote" Respository an.
```
git remote add <Name-remote-Repo> <url>
```
Beispiel (Verzeichnis `/git/pm1` auf dem Rechner unter `git.server.de`):
```
git remote add server ↵  git.server.de:/git/pm1
```

Stand lokales Repo ⟶ entferntes Repo
Um den aktuellen Stand des lokalen Repositories auf ein entferntes Repository zu senden, verwendet man das Kommando `push`:
```
git push <Name-remote-Repo> master
```
Beispiel:
```
git push server master
```
Hinweis: `master` ist der Name des Haupt-Zweiges. In einem Repo kann es mehrere Zweige geben. `master` ist der Standard.

Stand entferntes Repo ⟶ lokales Repo
Umkehrung von push:
```
git pull <Name-remote-Repo> master
```

Konflikte auflösen
Wurden gleichzeitig in unterschiedlichen lokalen Repositories Änderungen an der gleichen Datei vorgenommen, muss ein Konflikt aufgelöst werden. Git erstellt dann eine gemeinsame Version der beiden Dateiversionen (die oft kein gültiger Quellcode mehr ist). In dem Fall bereinigt man den Quellcode manuell, fügt die Datei neu in den *index* ein (`git add`) und sendet die Aktualisierung in das *head* (`git commit`)

Weitere Tutorials und Einführungen
- *http://rogerdudler.github.io/git-guide/*
- *http://mrchlblng.me/2014/09/practical-git-introduction/*
- *https://try.github.io/levels/1/challenges/1*
- *http://git-scm.com/book*