Git Einführung

Inhaltsverzeichnis

Einleitung

Beim Entwickeln von Software im Team stellen sich einige Fragen: Wo speichert man den Code, den man schreibt? Wie schafft man es, dass alle Entwickler Zugriff auf die aktuellste Version haben? uvm.

Git ist ein Programm, das uns diese Fragen beantwortet. Mithilfe von git kann man nicht nur **Code teilen**, sondern sogar **„in der Zeit zurückreisen“**.
Beispielsituationen: Mitten im Projekt unterläuft einem ein gravierender Fehler und man möchte alles wieder ungeschehen machen. Mit git kann man den Code **auf den Stand vor dem Fehler zurückbringen**.
Oder: Mitten im Projekt bemerkt man, dass sich zu einem unbekannten Zeitpunkt ein gravierender Fehler eingeschlichen hat. Mit git kann durch die Versionen suchen und den **Zeitpunkt finden, seitdem der Fehler existiert**.

Mit Git werden also Dateien in einem bestimmten Ordner (diesen bestimmt man bevor das Projekt beginnt) gesichert, im sogenannten **Projektordner**.

Git im Detail

Änderungen teilen

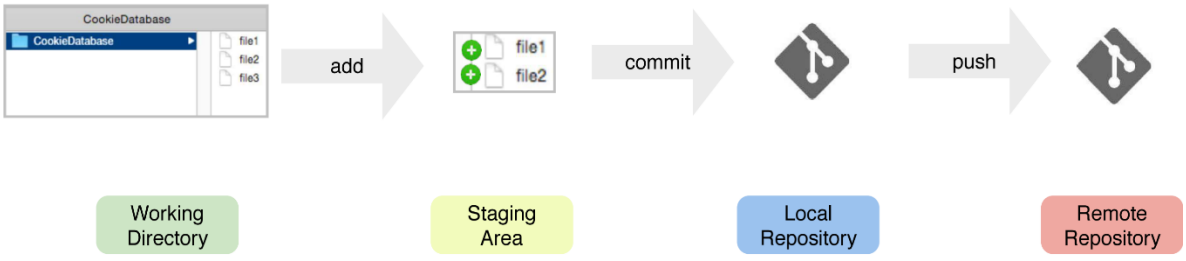

Abb. 1: Veränderungen am Code teilen

Im Grunde genommen hilft git den Entwicklern den Code (also die Dateien, in denen Code steht) **in einem Projektordner** intelligent und systematisch **abzuspeichern**.

Git ist konzeptuell in **vier <u>virtuelle</u> Speicherorte** eingeteilt:

1. *Working Directory*
    - Noch ungesicherte Änderungen (noch „frisch“ geschriebener Code)
    - Nicht für das ganze Team sichtbar
2. *Staging Area*
    - Zu sichernde Änderungen
      (kann z.B. auch nur ein Teil der Dateien sein; Änderungen können auch einfach ignoriert werden)
    - Nicht für das ganze Team sichtbar
3. *Local Repository*
    - gesicherte Änderungen
    - Nicht für das ganze Team sichtbar
4. *Remote Repository*
    - gesicherte Änderungen auf einem externen Server
    - Für das ganze Team sichtbar

Jede neue Änderung am Code **muss alle Speicherorte einmal durchlaufen**. Änderungen befördert man vom einen zum anderen Speicherort mit den Befehlen *add, commit, push*, wie oben in Abb. 1 skizziert.

<u>Beispiel</u>: Die Datei foo.java wurde erstellt.

1. Zuerst befindet sie sich im *working directory.*
2. Mit *add* wird vorgemerkt, dass foo.java gesichert werden soll. Nun befindet sich die Datei in der *staging area.*
3. Mit *commit* wird foo.java zunächst lokal gesichert. Im *local repository.*
4. Mit *push* wird foo.java auf den Server (*remote repository*) hochgeladen und kann nun von allen Teammitgliedern gelesen und bearbeitet werden.

Neue Änderungen (von anderen Teammitgliedern) laden

Neue Änderungen vom *remote repository* herunterzuladen ist einfach. Es braucht lediglich ein ***pull.*** Der Rest geschieht automatisch. Das Programm lädt alle verfügbaren Änderungen in den Projektordner, sodass das man wieder an der neuesten Version des Projekts arbeiten kann.

Installation

Hier in der Software AG benutzen wir **SourceTree**, ein weiteres Programm, um die Macht von git auszunutzen. Außerdem verwenden wir die Dienste von **Github**. Github stellt Server, auf denen man seine Dateien (kostenlos) speichern kann.

1. Account auf **github.com** erstellt.
2. Atlassian Account erstellen, um SourceTree verwenden zu können. (funktioniert auch mit Google-Account)
3. SourceTree herunterladen: https://www.sourcetreeapp.com/
4. Den Installationsschritten folgen

Fragen

1. Welche Probleme im Programmieralltag löst git?
2. Wie teilt man seine Änderungen, die man lokal am Code vorgenommen hat mit dem Team?
3. Wie checkt man, ob die Teammitglieder den Code geändert haben?
4. Wie heißt das Programm, das wir verwenden, um git zu benutzen?