UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR TELEMATIK

Übung zur Vorlesung „Werkzeuge für das wissenschaftliche Arbeiten – CS2450“, WS 17/18

Dozent: Dennis Boldt <boldt@itm.uni-luebeck.de>

Aufgabenblatt zum Thema git

Abgabe der Übung: Montag, 11.12.2017 bis 23:59 Uhr

Mit diesem Aufgabenblatt sollen die grundlegenden Kenntnisse von git erlernt werden.

Bearbeitungshinweise:

- Zum Bestehen dieses Aufgabenblattes müssen Sie die Links zu den Lösungen (d.h. github-Repositories) von Aufgabe 4, 5 und 6 ins Moodle laden. Die Aufgabe 4 beinhaltet die Ergebnisse aus Aufgaben 2 und 3.
- Die Abgabe erfolgt in 2er-Gruppen. Einzelabgaben werden nicht bewertet, außer nach vorheriger Rücksprache mit Dennis Boldt (bis spätestens 08.12.2017).
- Weitere Hinweise zur Abgabe finden sich am Ende dieses Aufgabenblattes.
- Lesen Sie sich jede Aufgabe **vor der Bearbeitung** einmal komplett durch.
- Wir empfehlen die Bearbeitung auf der Kommandozeile (Shell/Terminal/...).
- Den Vortrag finden Sie unter:

  `https://www.dennis-boldt.de/git`

Aufgabe 0: Vorbereitung

Zur Vorbereitung auf dieses Aufgabenblatt bearbeiten Sie bitte die beiden folgenden Online-Tutorials:

a) Einführung in git:

`https://try.github.io/`

b) Einführung in Markdown:

`https://www.markdowntutorial.com/`

Aufgabe 1: Einrichtung

Installation

- Sollten Sie dieses Aufgabenblatt auf den Poolrechnern bearbeiten, dann kann diesen Aufgabenteil übersprungen werden.
- Sollten Sie dieses Aufgabenblatt auf Ihrem eigenen Rechner bearbeiten, dann installieren Sie bitte git[1].

Konfiguration

- Konfigurieren Sie Ihren globalen Benutzernamen und Ihre globale E-Mail-Adresse (`git config`).
- Konfigurieren Sie farbige Kommandozeilen-Ausgaben.
- Schauen Sie sich die globale Konfigurationsdatei an (`~/.gitconfig`).

Github

- Im Rahmen dieses Aufgabenblattes wollen wir `github.com` verwenden, damit Sie von Anfang an den Umgang damit lernen. Sollten Sie bereits einen Account bei Github haben, so können Sie diesen Aufgabenteil überspringen.
- Wir empfehlen sich bei der Wahl des Benutzernamen Gedanken zu machen, sodass Sie diesen Account auch für weitere Tätigkeiten bei Github benutzen können, z.B. dem Melden oder Fixen von Issues anderer Projekte.
- Erstellen Sie einen Account unter:

  `https://github.com/`

[1] `https://git-scm.com/book/en/v2/Getting-Started-Installing-Git`

SSH-Keys

- Erstellen Sie ein SSH-Schlüsselpaar. Sorgen Sie dafür, dass der **private Schlüssel** immer bei ihnen bleibt. Laden Sie den **öffentlichen Schlüssel** bei Github hoch (`ssh-keygen`).

Aufgabe 2: Local Repository

a) Erstellen Sie ein leeres Verzeichnis (Working Tree).

b) Erstellen Sie in diesem Verzeichnis ein Local Repository (`git init`).

c) Schauen Sie sich dieses Local Repository (`.git`-Verzeichnis) näher an. Bitte nichts daran verändern!

d) Erstellen Sie im Working Tree eine leere Datei Namens `README.md`.

e) Fügen Sie diese leere Datei zum Index hinzu (`git add`).

f) Schauen Sie sich mittels `git status` den Lebenszyklus der Dateien an.

g) Erstellen Sie einen initialen Commit, welcher die Datei aus dem Index beinhaltet (`git commit`)

h) Formatieren Sie die `README.md` mit Markdown[2]. Die `README.md` sollte Überschriften in verschiedenen Größen und mindestens eine unsortierte Liste enthalten. Für Dummy-Texte können Sie einen *Lorem Ipsum Generator* verwenden[3].

i) Erstellen Sie einen weiteren Commit, welcher das hinzugefügte Markdown beinhaltet.

j) Schauen Sie sich mittels `git log` die History an. Probieren Sie dabei verschiedene Parameter (u.a. `-1` oder `--oneline`) aus.

Aufgabe 3: Lokale Branches

a) Erstellen Sie einen Branch mit dem Namen `dev` und wechseln Sie auf diesen (`git branch` und `git checkout`).

b) Erstellen Sie auf dem Branch `dev` eine weitere Datei mit dem Namen `INSTALL.md`, welche Sie auch mit Markdown füllen.

c) Fügen Sie die `INSTALL.md` in einem Commit zum Branch `dev` hinzu.

d) Wechseln Sie zurück zum Branch `master`.

e) Erstellen Sie auf dem Branch `master` eine weitere Datei mit dem Namen `LICENSE.md`, welche Sie mit einem *Lorem Ipsum* füllen.

[2] `https://guides.github.com/features/mastering-markdown/`
[3] `http://www.loremipsum.de/` oder `http://lipsum.com/`

f) Fügen Sie die `LICENSE.md` in einem Commit zum Branch `master` hinzu.

g) Mergen Sie den Branch `dev` auf den Branch `master` (`git merge`). Überlegen Sie vorher, um welche Merge-Strategie es sich dabei handelt. Überprüfen Sie, ob Sie Recht haben.

Aufgabe 4: Remote Repository

a) Erstellen Sie auf Github ein neues Repository mit dem Namen `werkzeuge-aufgabe4`.

b) Fügen Sie dieses Remote Repository zu Ihrem lokalen Repository hinzu (`git remote`).

c) Pushen Sie den Branch `master` in das Remote Repository (`git push`).

d) Pushen Sie den Branch `dev` in das Remote Repository.

e) Öffnen Sie Ihr Repository auf Github im Browser und überprüfen Sie, ob beide Branches vorhanden sind, und ob die `README.md`, `INSTALL.md` und `LICENCE` richtig formatiert sind. Falls nicht, korrigieren Sie das mit einem neuen Commit.

f) Erstellen Sie einen Tag `aufgabe4` (`git tag`).

g) Pushen Sie diesen Tag zu Github.

h) Öffnen Sie das Repository im Browser und überprüfen Sie, ob der Tag vorhanden ist.

Aufgabe 5: Merge Conflict

a) Forken Sie auf Github das folgende Repository:

`https://github.com/boldt/werkzeuge-aufgabe5`

b) Klonen Sie über SSH das eigene geforkte Repository (`git clone`).

c) Dieses Repository beinhaltet zwei Branches: `master` und `bugfix`. Beide beinhalten ein HelloWorld-Programm in Java.

d) Wechseln Sie in den Branch `bugfix` und schauen Sie sich das HelloWorld-Programm an (`git checkout`).

e) Wechesln Sie zurück in den Branch `master`.

f) Mergen Sie den Branch `bugfix` in den Branch `master` (`git merge`).

g) Bei diesem Merge wird es zu einem Konflikt kommen. Lösen Sie den Konflikt, sodass das Java-Programm lauffähig ist und dabei `Moin Moin` ausgibt. Testen Sie dies mit Hilfe von `javac` und `java`.

h) Erstellen Sie eine `README.md`, welche kurz beschreibt, wie man das Programm kompiliert und ausführt.

i) Erstellen Sie eine `.gitignore`, um die vom Java-Compiler erstellten Dateien vor dem einchecken zu schützen.

j) Fügen Sie die `.gitignore` als neuen Commit hinzu.

k) Pushen Sie Ihre Änderungen in das von Ihnen geforkte Repository.

l) Erstellen Sie einen Tag `aufgabe5`.

m) Pushen Sie diesen Tag zu Github.

Aufgabe 6: Zurücksetzen

a) Klonen Sie über SSH das folgende Repository (nicht forken!):

`https://github.com/boldt/werkzeuge-aufgabe6`

b) Erstellen Sie ein neues, leeres Repository bei Github mit dem folgenden Namen: `werkzeuge-aufgabe6`

c) Ändern Sie den Remote Branch `origin` in Ihrem Local Repository auf Ihr neu erstelltes Remote Repository. **Hinweis**: Bitte noch nicht pushen.

d) Verwenden Sie `git log` und `git diff`, um den folgenden Commit zu finden:

- In dem die `README.md` zum Projekt hinzugefügt wurde.

e) Setzen Sie die History **vor diesen Commit** (hart) zurück.

f) Pushen Sie jetzt in das von Ihnen zurückgesetzte Repository.

g) Erstellen Sie einen Tag `aufgabe6`.

h) Pushen Sie diesen Tag zu Github.

Hinweise zur Abgabe

Bearbeiten Sie diese Aufgabe in Zweiergruppen. Sie sollten jetzt in Ihrem Github-Acccount drei Repositories haben:

- `werkzeuge-aufgabe4`,
- `werkzeuge-aufgabe5` und
- `werkzeuge-aufgabe6`.

Ein Student der Zweiergruppe lädt die Links zu den drei Repositories ins Moodle. Nennen Sie den Namen und die Matrikelnummern beider Studenten der Zweiergruppe. Sollten beide Studenten Lösungen hochladen, so werden wir nur eine zufällig gewählte bewerten.