

Übungen zur Vorlesung Softwaretechnologie

- Wintersemester 2014/15 -

Dr. Günter Kiesel

Übungsblatt 1

Zu bearbeiten bis: 26.10.2014

Fragen zu Übungsaufgaben respektive zur Vorlesung können Sie auf der Mailingliste swt-tutoren@lists.iai.uni-bonn.de, bzw. swt-vorlesung@lists.iai.uni-bonn.de stellen.

Aufgabe 1. *Branch & merge sowie vollständig dezentralisiertes Arbeiten* (13-18 Punkte)

- a) (1 Punkt) Führen Sie ein "pull" von dem in Blatt 0 eingerichteten GitHub-Repository durch.

Danach sollen Sie in Ihrem lokalen git zusätzlich zu vorher die Datei "briang_kellogg.txt" vorfinden. Darin steht ein Text der aus drei Abschnitten besteht.

Jeder von Ihnen soll *parallel zu den anderen Gruppenmitgliedern* auf seinem eigenen Rechner genau einen Abschnitt bearbeiten. Ordnen Sie sich selbst die Abschnitte zu, indem Sie ihre Vornamen alphabetisch sortieren. Der 1. Vorname bearbeitet den 1. Abschnitt, der 2. Vorname den zweiten Abschnitt, und so weiter. Wenn Sie mehr als 3 Teammitglieder sind, geht es zyklisch weiter. Somit gibt es mehrere Leute, die den gleichen Abschnitt bearbeiten.

- b) (5 Punkte) Erstellen Sie nun für sich einen eigenen Branch, wechseln Sie zu dem neuen Branch ("switch") und führen Sie darin folgende Änderungen durch:

1. Korrigieren Sie die enthaltenen Rechtschreibfehler in der Datei *briand-kellogg.txt* entsprechend Ihrer vereinbarten Aufgabenteilung pro Abschnitt.
2. Benennen Sie die Datei *briand-kellogg.txt* in *kellogg-briand.txt* um.
3. Committen Sie die Änderungen in ihr lokales Repository.

- c) (1 Punkt) Nun nehmen wir mal an, dass die Verbindung zum GitHub-Server gerade nicht verfügbar ist, Sie aber trotzdem mit Unterstützung von Git mit Ihren Kollegen weiterarbeiten wollen. Übergeben Sie dazu ihren lokalen Git-Ordner ihre Teamkollegen (z.B. als Kopie via USB-Stick oder als per E-Mail versendetes zip-Archiv des gesamten Ordners). Tip: Der Git-Ordner ist der Ordner in den sie das geklonte repository gespeichert haben. Darin sollte ein ".git" Unterverzeichnis vorhanden sein, das das eigentliche Repository darstellt. Was Sie sonst noch darin sehen ist der aktuelle Zustand Ihres lokalen Arbeitsbereiches. Übergeben Sie Ihren Kollegen den *gesamten* Git-Ordner, d.h. Repository *und* Arbeitsbereich.

- d) (5 Punkte) Mergen Sie den Branch eines ihrer Kollegen auf ihren eigenen Branch.
- Dazu müssen Sie zuerst das von einem Kollegen zugesendete Repository lokal speichern und unter *Remotes* → *Manage Remotes* als Bezugsrepository hinzufügen.

- Anschließend von diesem Remote ein Pull durchführen.
- Eventuell auftretende Konflikte zeigt Ihnen SmartGit durch ein Dateisymbol mit einem roten Punkt in dem sich ein Ausrufezeichen befindet. Lösen Sie die Konflikte auf, indem Sie (in SmartGit) auf die entsprechende Datei doppelklicken. In dem sich öffnenden Fenster können Sie alle Änderungen der Datei nachverfolgen und die Konflikte auflösen indem Sie pro Änderung entscheiden, welcher Stand (ihrer oder der des Anderen) übernommen wird. Sprechen Sie gegebenenfalls mit ihrem am Konflikt beteiligten Kollegen, wenn unklar ist, welche Konfliktlösung Sinn macht. Schauen Sie sich im Konfliktlösungseditor genau um, so dass Sie im Tutorium erklären können, wann genau ein Konflikt auftritt. Versuchen Sie insbesondere zu verstehen welche der folgenden Fälle zutreffen:
 - i. Wenn die gleiche Datei von unterschiedlichen Leuten bearbeitet wurde, gibt es immer einen Konflikt.
 - ii. Wenn die gleiche Zeile einer Datei von unterschiedlichen Leuten bearbeitet wurde, gibt es einen Konflikt.
 - iii. Wenn in der gleichen Zeile von unterschiedlichen Leuten unterschiedliche Änderungen durchgeführt wurden gibt es einen Konflikt.
 - iv. Wenn ... *<Ihre eigene Formulierung, falls keine der obigen passt>* ...
- Wenn alle Konflikte in der Datei gelöst sind speichern Sie. Wenn Sie dabei danach gefragt werden, ob für die die Konfliktlösung ein "Stage" durchgeführt werden soll antworten Sie mit "Ja". Anschließend committen Sie.

Vereinbaren Sie, wer in ihrem Team als "Integrator" für die finale Version verantwortlich ist. Die Punkte zu f) und g) gibt es natürlich nur für den Integrator.

- e) (1 Punkt) Schicken Sie ihren Git-Ordner mit dem Zustand nach d) an den Integrator.
- f) (1 Punkt pro branch den Sie integrieren) Der Integrator hat die Aufgabe alle Branches in den 'Master'-Branch zu mergen.
- g) (1 Punkt) Abschließend schickt er seinen Git-Ordner als ZIP-Datei gepackt als Gruppenabgabe seinem Tutor per mail.

Tip: Natürlich ist es nicht sinnvoll, nur um einige Änderungen zu Teilen ein ganzes Repository zu verschicken, insbesondere wenn es eine nicht-triviale Größe hat. Daher gibt es die Möglichkeit, einen patch zu erstellen und per mail zu versenden. Das braucht wenig Datentransfer und bietet trotzdem volle git-Unterstützung (Merge, Historie, ...). Siehe z.B:

- <http://ariejan.net/2009/10/26/how-to-create-and-apply-a-patch-with-git/>
- http://wiki.tuxbox.org/Development:Git_Patch_erstellen
- <http://git-scm.com/docs/git-format-patch>

So würde man in der Praxis vorgehen, wenn Sie mit jemandem zusammen arbeiten wollen, ohne dass Sie auf sein oder er auf Ihr Repository zugreifen kann (was bei Open-Source-Entwicklung der Normalfall ist). Wir haben es hier nicht geübt um die Aufgabe nicht noch komplexer zu machen.

Tip 2: Manche Werkzeuge (z.B. Eclipse Egit) unterstützen die Patch-Erzeugung auch in der GUI (SmartGit leider nicht).

Aufgabe 2. *Dateioperationen mit Git* (8 Punkte)

Hier üben wir noch mal das komplett dezentrale Arbeiten (ohne Git-Server) und nehmen diesmal das Erzeugen, Löschen und Umbenennen von Dateien mit hinzu. Mit einem zentralen Server von dem Sie direkt clone / pull / push durchführen, würde alles genauso funktionieren (nur das Versenden / Kopieren der Repositorien und Umsetzen des Remote würde entfallen).

- a) (1 Punkt) Erzeugen sie lokal ein neues Projekt „Dateien“ an und legen Sie darin einen Ordner „Nummer 1“ mit einer Datei „Datei 1.txt“ an.
- b) (1 Punkt) Erzeugen Sie dafür lokal ein Git-Repository (SmartGit: *Repository* → *Add or Create...*).
- c) (1 Punkt) Erzeugen Sie nun einen Branch den Sie mit ihrem Vornamen benennen und wechseln Sie in diesen Branch
- d) (1 Punkt) Fügen Sie (auf Systemebene) in dem Ordner eine neue Datei „Datei 2.txt“ hinzu. Wechseln Sie nach SmartGit. Sehen Sie die neue Datei? Sehen Sie die alte Datei auch noch? Finden Sie heraus worüber Sie steuern können, ob sie alle Dateien / neue Dateien / veränderte Dateien sehen.
- e) (1 Punkt) Welches Symbol hat die neue Datei? Teilen Sie git mit, dass sie auch ins Repository übernommen werden soll, indem Sie darauf ein "Stage" durchführen. Wie verändert sich das Dateisymbol?
- f) (1 Punkt) Schreiben Sie die Antworten zu d und e in die erste Datei. Wie verändert sich das Dateisymbol nach dem Speichern? Schreiben sie auch die Antwort hierzu in die Datei.
- g) (1 Punkt) Führen Sie nun einen Commit durch.
- h) Benennen Sie nun (auf Systemebene) „Datei 2.txt“ in „Datei 2a.txt“ um. Was bemerken Sie?
- i) (1 Punkt) Wie lässt sich das erklären? Schreiben Sie es in die Datei und führen Sie einen finalen commit durch.
- j) Schicken sie das Ergebnis Ihrem Tutor (den kompletten git-Ordner als zip).

Hinweis: Verwenden Sie zur Bearbeitung der Aufgaben auch weiterführende Literatur bzw. Web-Suchanfragen.