**Git & Spike Solutions**

Eine Einführung

von

Marco Egner, Dominik Thirmeyer

1. Das Versionskontrollsystem Git

Git ist ein Open Source Versionskontrollsystem, das unter der General Public License in der Version 2 veröffentlicht wurde.



Abbildung 1: Git Logo

Der Software Client steht für folgende Systeme bereit:

* Windows
* Linux
* Solaris
* Mac OS X

2. Funktionsweise von Git

2.1 Dateiversionierungsverfahren unter Git

Anders als viele Versionskontrollsysteme, verwaltet Git die Änderungshistorie nicht mittels Änderungslisten von einzelnen Dateien, sondern legt jeweils einen Snapshot an. Konkret heißt dies, es wird bei jedem Commit ein Speicherabbild (Snapshot) erstellt, das alle Daten enthält. Um zu große Datenmenge zu vermeiden, werden für Dateien die keine Modifikationen erfahren haben, lediglich Referenzen auf die vorherige Version erzeugt.

2.2 Die drei Zustände des lokalen Git-Repository

Das Git Repository besteht aus dem Arbeitsverzeichnis (working directory), welches einem Abbild (checkout) einer spezifischen Version des Projektes entspricht, der Staging Area, die alle für den nächsten Commit vorgemerkten Dateien enthält und dem eigentlichen Git Repository, das sämtliche Speicherabbilder des Projekts enthält.

Ausgehend von einem Arbeitsverzeichnis, werden noch nicht versionierte Dateien und veränderte bereits versionierte Dateien mittels *git add <dateiname>* in die Staging Area übernommen. Damit diese Veränderungen nun in das Git Repository eingefügt werden, wird ein *git commit –m „<CommitNachricht>“* ausgeführt. Bei einem verteilten Repository ist noch ein git push nötig um dieses zu synchronisieren.

2.3 Branches

Branches werde in Git angelegt, um eigene Entwicklungsstränge bereitzustellen.

Dies erleichtert das Arbeiten mit Tickets bzw. die Arbeitsweise in agilen Projekten.

Um einen neuen Branch anzulegen, wird der Befehl *git checkout –b <newBranchName>* angewandt. In diesem können nun die Erweiterungen und Anpassungen vorgenommen werden, die den jeweiligen Arbeitsschritt im Projekt entsprechen.

Soll die Modifikation nun in das Projekt mit einfließen, so wird der neue Branch mit dem master-Branch zusammengeführt, dies geschieht mit Hilfe von *git merge <master>*.

3. Die wichtigsten Befehle in Git

* **git init**

Erstellt ein neues Git-Repositry

* **git clone /pfad/zum/repository oder**

**/pfad/zum/repository <Ordnername>**

Erstellt eine Arbeitskopie des Git Repository, gegebenenfalls in extra angegebenen Ordner

* **git add <dateiname> oder git add \***

Fügt eine, mehrere oder alle noch nicht aufgeführten Dateien zum Repository hinzu oder merkt bereits vorhandene und modifizierte Dateien für den nächsten Commit vor

* **git rm <dateiname>**

Löscht die Angegebene Datei aus dem Repository (bei ganzen Verzeichnissen wird der Parameter –r benötigt)

* **git commit –m “<Commit Nachricht>“**

Bestätigen der Änderungen (modifizierte, hinzugefügte und gelöschte Dateien) gegenüber des lokalen Git Repository

* **git status**

Zeigt die getätigten Änderungen (nicht commited) an

* **git push**

Übertragen der Commits an das Master Repository im aktuellen Branch

* **git pull**

Aktualisiert das lokale Repository mit den neuesten Änderungen

* **git diff <quell\_branch> <ziel\_branch>**

Zeigt die Differenzen zweier Branches (vergleichbare Syntax bei commits)

4. Spike Solution

Unter dem Begriff Spike Solution versteht man kurz gesagt ein Code Experiment.

Ziel hierbei ist es, Wissen durch einfaches Ausprobieren möglichst schnell zu gewinnen. Dabei achtet man nicht auf Clean-Code-Constrains sowie Unit-Tests und schreibt dabei nur so viel Code wie nötig um seine definierte Fragestellung zu beantworten.

Eine gängige Praxis ist es, Spikes in eigenen Branches zu kapseln. Der darin erzeugte Quelltext wird aber keinesfalls in das Projekt übernommen, das heißt der Branch wird nicht per merge-Befehl mit dem master-Branch zusammengeführt.

Ist die Frage beantwortet, wird der Code nicht verbessert, sondern weggeworfen. Das aus dem Experiment gewonnenen Wissen wird dazu genutzt, um eine saubere, den Richtlinien entsprechende Implementierung zu erzeugen.

Quellen:

<https://git-scm.com/book/de/v1>

<https://git-scm.com/docs>

<http://qualitycoding.org/spike-solution/>