

Inhaltsverzeichnis

1. Einleitung	7
1.1 Versionsverwaltung – Was ist das?	9
1.2 Die Entstehung von Git	12
1.3 Was zeichnet Git aus?	15
2. Die Vorbereitungsmaßnahmen	20
2.1 Der Kommandozeileninterpreter: wichtiges Hilfsmittel für die Arbeit mit Git.....	20
2.2 Git auf verschiedenen Systemen installieren	23
2.3 Die grundlegende Konfiguration.....	26
2.4 Wenn es einmal nicht weitergeht: die Git-Hilfe	31
2.5 Übungsaufgaben.....	34
3. Die Grundlagen für die Arbeit mit Git	37
3.1 Das Git-Repository – Ausgangspunkt für die Verwendung von Git.....	37
3.2 Dateien zum Repository hinzufügen	41
3.3 Dateien verändern, löschen und umbenennen.....	48
3.4 Änderungen rückgängig machen und Commits ergänzen	55
3.5 Den Verlauf der Commits anzeigen	58
3.6 Frühere Versionen wiederherstellen	62
3.7 Markierungen mit Tags.....	66
3.8 Übungsaufgaben.....	72
4. Branches mit Git erstellen	76
4.1 Branching: eine besondere Stärke von Git	76
4.2 Branches erstellen und wieder zusammenführen.....	77
4.3 Die Verwaltung von Branches.....	84
4.4 Zweige mit dem rebase-Befehl zusammenführen.....	86
4.5 Übungsaufgaben.....	94
5. GitHub: Plattform für die Zusammenarbeit über das Internet	98
5.1 Ein kostenloses Konto bei GitHub erstellen	99
5.2 Ein eigenes Projekt bei GitHub erstellen.....	101
5.3 Möglichkeiten für die Zusammenarbeit über GitHub	113
5.4 Remote-Repositories für die Zusammenarbeit über das Internet.....	118
5.5 Remote-Branches.....	127
5.6 Ein lokales Verzeichnis auf GitHub übertragen	132
5.7 Verschiedene Alternativen zu GitHub	133
5.8 Übungsaufgaben.....	136

6. Customizing: Git an die eigenen Anforderungen anpassen	140
6.1 Weiterführende Konfigurations- und Anpassungsmöglichkeiten	140
6.2 Aliasse für eine einfachere Arbeit mit Git	143
6.3 Attribute	146
6.4 Hooks	150
6.5 Übungsaufgaben.....	157
7. Fortgeschrittene Funktionen von Git	160
7.1 Inhalte in Git suchen	160
7.2 Eine passende Revision auswählen.....	165
7.3 Den Verlauf nachträglich bereinigen	170
7.4 Für eine eindeutige Zuordnung: Beiträge signieren	175
7.5 Fortgeschrittene Funktionen für das Staging	180
7.6 Für Ordnung sorgen: stash und clean.....	185
7.7 Fortgeschrittene Funktionen für das Merging	192
7.8 Die Funktion rerere	197
7.9 Git für die Fehlerbehebung verwenden.....	200
7.10 Übungsaufgaben.....	204
8. Git auf einem Server verwenden	208
8.1 Die Auswahl eines passenden Protokolls.....	208
8.2 Ein Repository auf einem Server bereitstellen.....	212
8.3 SSH-Schlüssel hinzufügen	215
8.4 Den Git-Daemon einrichten	216
8.5 Mit Smart HTTP arbeiten.....	219
9. Verteilte Systeme	226
9.1 Unterschiedliche Organisationsformen für die Zusammenarbeit.....	226
9.2 An einem gemeinsamen Projekt mitarbeiten.....	229
9.3 Ein Projekt selbst verwalten	235
9.4 Übungsaufgaben.....	238
10. Mit Submodulen arbeiten	242
10.1 Was ist ein Submodul und wozu dient es?	242
10.2 Ein Submodul in ein Projekt einfügen	244
10.3 Projekte mit Submodulen klonen	246
10.4 Die Verwendung von Submodulen	249
10.5 Übungsaufgaben.....	254
11. Die internen Abläufe von Git	257
11.1 Unterschiedliche Befehlskategorien	257
11.2 Was ist ein Git-Objekt?	259
11.3 Referenzen in Git und Referenzspezifikationen	264

11.4	Packdateien für eine effiziente Datenspeicherung.....	268
11.5	Dateien wiederherstellen.....	270
11.6	Übungsaufgaben.....	272
12.	Git im Zusammenspiel mit anderen Systemen für die Versionsverwaltung	275
12.1	Subversion und andere Systeme für die Versionsverwaltung.....	275
12.2	Git als Client für andere Systeme verwenden.....	277
12.3	Zu Git migrieren.....	281
13.	Git gemeinsam mit anderen Entwicklungswerkzeugen verwenden	285
13.1	Die praktische Anwendung von Git bei der Programmerstellung.....	285
13.2	Tools für eine einfachere Anwendung in der Shell.....	286
13.3	Die Integration von Git in verschiedene integrierte Entwicklungsumgebungen.....	287
13.4	Grafische Benutzeroberflächen für Git verwenden.....	292
14.	Fazit und Ausblick	298
15.	Glossar	302
16.	Stichwortverzeichnis	306