

# Inhalt

<b>Vorwort</b> .....	<b>VII</b>
<b>1 Einführung</b> .....	<b>1</b>
Hintergrund .....	1
Die Geburt von Git .....	2
Die Vorgeschichte .....	5
Der weitere Verlauf .....	6
Namen sind Schall und Rauch? .....	7
<b>2 Git installieren</b> .....	<b>9</b>
Die Linux-Binärdistributionen benutzen .....	9
Eine Quellversion beschaffen .....	11
Kompilieren und Installieren .....	12
Git unter Windows installieren .....	13
<b>3 Erste Schritte</b> .....	<b>19</b>
Die Git-Kommandozeile .....	19
Kurze Einführung in die Benutzung von Git .....	21
Konfigurationsdateien .....	28
Ausblick .....	31
<b>4 Grundlegende Git-Konzepte</b> .....	<b>33</b>
Grundlegende Konzepte .....	33
Darstellungen des Objektspeichers .....	38
Git-Konzepte am Werk .....	41

III

<b>5</b>	<b>Dateiverwaltung und der Index</b>	<b>51</b>
	Es geht um den Index	52
	Dateiklassifikationen in Git	52
	git add benutzen	54
	Einige Hinweise zur Benutzung von git commit	56
	git rm benutzen	58
	git mv benutzen	60
	Ein Hinweis zum Überwachen des Umbenennens	62
	Die Datei .gitignore	63
	Eine ausführliche Darstellung des Objektmodells und der Dateien von Git	65
<b>6</b>	<b>Commits</b>	<b>71</b>
	Atomare Änderungsmengen (Changesets)	72
	Commits festlegen	73
	Der Commit-Verlauf	78
	Commits suchen (und finden)	90
<b>7</b>	<b>Zweige</b>	<b>97</b>
	Gründe für den Einsatz von Zweigen	97
	Zweignamen	99
	Zweige benutzen	100
	Zweige erzeugen	101
	Zweignamen auflisten	102
	Zweige anschauen	103
	Zweige auschecken	105
	Zweige löschen	112
<b>8</b>	<b>Diffs</b>	<b>115</b>
	Formen des git diff-Befehls	116
	Ein einfaches git diff-Beispiel	120
	git diff und Commit-Bereiche	124
	git diff mit Pfadbegrenzung	126
	Wie Subversion und Git Diffs ableiten – ein Vergleich	128
<b>9</b>	<b>Merges</b>	<b>131</b>
	Merge-Beispiele	132
	Mit Merge-Konflikten arbeiten	138
	Merge-Strategien	148
	Wie Git über Merges denkt	158

<b>10 Commits verändern</b>	<b>163</b>
Vorsicht beim Verändern des Verlaufs	165
git reset benutzen	166
git cherry-pick benutzen	174
git revert benutzen	176
reset, revert und checkout	176
Das oberste Commit ändern	178
Rebasing von Commits	180
<b>11 Entfernte Repositories</b>	<b>193</b>
Repository-Konzepte	194
Auf andere Repositories verweisen	198
Ein Beispiel mit entfernten Repositories	202
Entfernte Repository-Operationen in Bildern	214
Entfernte Zweige hinzufügen und löschen	221
Remote-Konfiguration	222
Bare-Repositories und git push	224
Repositories veröffentlichen	226
<b>12 Repository-Verwaltung</b>	<b>233</b>
Die Repository-Struktur	233
In einer verteilten Entwicklung leben	237
Wissen, wo Sie stehen	240
Mit mehreren Repositories arbeiten	245
<b>13 Patches</b>	<b>253</b>
Wozu Patches benutzen?	254
Patches generieren	255
Patches per E-Mail verschicken	264
Patches anwenden	267
Schlechte Patches	274
Patching und Merging im Vergleich	274
<b>14 Hooks – Einstiegspunkte für die Automatisierung</b>	<b>277</b>
Hooks installieren	279
Verfügbare Hooks	282
<b>15 Projekte kombinieren</b>	<b>289</b>
Die alte Lösung: partielle Checkouts	290
Die offensichtliche Lösung: den Code in Ihr Projekt importieren	292

Die automatisierte Lösung: Teilprojekte mit eigenen Skripten auschecken . . .	299
Die native Lösung: Gitlinks und git submodule . . . . .	301
<b>16 Git mit Subversion-Repositories benutzen . . . . .</b>	<b>309</b>
Beispiel: ein einfacher Klon eines einzigen Zweigs . . . . .	309
Verschieben, Ziehen, Verzweigen und Zusammenführen mit git svn . . . . .	316
Verschiedene Hinweise zum Arbeiten mit Subversion . . . . .	323
<b>Index . . . . .</b>	<b>327</b>