



기호

!(느낌표), 역 패턴	73	.dotest 디렉터리	339
#(파운드 기호)	72	.git 디렉터리, SVN 변환을 위한 메타데이터	456
.gitignore 파일	72	.git 하위 디렉터리	46-48
주석	157	.git/config 파일	243, 246, 274, 33, 38, 147
#include 문	437	.gitmodules 파일 설정 복사하기	377-378
\$GIT_DIR 변수	83	브랜치 항목 추가하기	279
--(2중 대시) 피연산자 구분 기호	23	추가하기	276, 277
---(3중 빼기 기호)		.git/hooks 디렉터리	347, 349
diff	131	.git/logs 디렉터리, 저장된 reflog	237
- (빼기 기호)		.git/logs/HEAD 파일	237
diff	131, 162	.git/MERGE_HEAD 파일	165
git show-branch 출력	116	.git/MERGE_MSG 파일	165
분리 작업에서의 위치	437	.git/objects 디렉터리	47
.(작은 따옴표), 한 단어로 참조	236	.git/rebase-apply 디렉터리	339
*(별표)		.git/refs/ 디렉터리	83
git show-branch 출력	116	.gitignore 파일	60, 61, 72-74
글로벌링	72	svn:ignore와 비교	408
브랜치 이름 규칙	111	.gitmodules 파일	377
.(점)		.git/config 파일에 설정 복사하기	377-378
브랜치 이름 규칙	111	기록된 주소의 일부로서의 사용자 이름	387
현재 디렉터리	25	.NET 관련 제품, 어셈블리(Assemblies)	383
..(2중 점)		.o 파일, 무시하기	73
커밋 범위	95	.sample 접미사, 실행 가능한 후크 이름	350
git diff 명령	141	.svn 디렉터리	393
...(3중 점), 대칭 차집합	99, 144, 282	/(슬래시 문자), 후미, 디렉터리 이름	72, 111, 374
		/etc/gitconfig 파일	33
		/etc/inetd.conf 파일	291

/etc/xinetd.d/git-daemon 파일	291
:(콜론)	
refspec	248
브랜치 이름 규칙	111
?(물음표), 브랜치 이름 규칙	111
@(at 기호)	
reflog	196, 233
스태시 항목 이름	225, 230
[](대괄호), git show-branch 출력	116
^(캐럿)	84-87
브랜치 이름	117
브랜치 이름 규칙	111
~(물결 기호)	84-87
브랜치 이름 규칙	111
~/,gitconfig 파일	33
+(더하기 기호)	
diff	131, 162
git show-branch 출력	116
refspec	248
분리 작업에서의 위치	437
커밋 차이점을 볼 때	30
+++ (3중 더하기 기호), diff	132
<<(왼쪽), git log --left-right 표시	164
<<<<<<	
diff 출력	156
병합 표시	160, 180
>(오른쪽), git log --left-right 표시	164
>>>>>>	
diff 출력	156
병합 표시	160, 180
0이 아닌 상태, 프리 후크	348
-3 또는 --3way 옵션	
git am 명령	342

A

-a 옵션	
git branch 명령	259
git diff 명령	139

--abbrev-commit 옵션, git log 명령	88, 153, 229, 393
--abort 옵션	
git am 명령	338
git rebase 명령	209
add	git add 명령 참조
--all 옵션	
git commit 명령	64
git filter-branch 명령	454
am	git am 명령 참조
--amend 옵션, git commit 명령	22, 203
Apache 구성 파일	294
applypatch-msg 후크	355
approximate() 함수	424-425
apt-get 명령	10

B

Bash(명령 프롬프트)	16
bisect	git bisect 명령 참조
BitKeeper VCS	2, 6
blame	git blame 명령 참조
blob 객체	38-39, 40, 48
참조되지 않은	451
branch.branch_name.rebase 구성 변수	264

C

--cached 옵션, git diff 명령	134
CAS(Central Authentication Service)	506
cat 명령, 구성 파일 콘텐츠 보기	34
cat-file	git cat-file 명령 참조
Central Authentication Service(CAS)	506
cherry-pick	git cherry-pick 명령 참조
chmod 755 명령	293
commit-filter	413
commit-msg 후크	349, 354
config	git config 명령 참조
--continue 옵션, git rebase 명령	208

core.logAllRefUpdates 부울 구성 옵션	233
criss-cross 병합	172, 176, 179
curl-config 도구	13
cvs update 명령	122
CVS(Concurrent Version System)	6, 361
프로젝트로 코드 가져오기, 단점	362

D

DAG(directed acyclic graph)	91
그리기	94
dcommit	401, 406, 409
diff 프로그램	30, 507, git diff 명령 참조
-r 옵션	132
Diff	131-148
병합 커밋	168
서브버전과 Git 비교	147
DISPLAY 환경 변수	105
dofile 구성	388
--dry-run 옵션, git commit 명령	225, 231

E

Emacs Lisp 설정	388
emerge 명령(젠투)	11
env-filter	412
DISPLAY	105
GIT_AUTHOR_NAME	27
expat.h 도구	13
externals 스크립트, 마일스 조지(Miles Georgi)	372

F

fetch	git fetch 명령 참조
FETCH_HEAD	84
--file 옵션	33
filter-branch	git filter-branch 명령 참조
find 명령	46

--first-parent 옵션	183
--follow 옵션	466
git log 명령	70
format-patch	git format-patch 명령 참조
Freedesktop.org, 개발	302

G

gc.auto 매개변수	453
gc.autopacklimit 매개변수	453
gc.pruneExpire 매개변수	129, 453
gc.reflogExpire 매개변수	129, 237, 453
gc.reflogExpireUnreachable 매개변수	237, 453
Git	9, 13-14
Git 설치하기	Git 설치하기 참조
contrib 디렉터리, 예제 후크	351
--recurse-submodules 옵션	389
SVN 저장소와 함께 사용하기	391-409
--version	12
구성 파일	33-35
날짜 구문 분석	424
명령행	21-23
소스 저장소	278
소스의 마스터 저장소	12Git
코딩 모델	503-505
콘텐츠 추적 시스템	40
타임라인	7
탄생	2-5
파일 분류	59-61
git add 명령	25, 49, 59, 61-69 157, 364
gitlink 업데이트	379
--interactive 옵션	63
-p 옵션, 대화식으로 형크 준비하기	429, 432
파일 충돌 상태 해결하기	167
효과	76
git add git 명령	366
git am 명령	309, 322, 337, 338, 354
--abort 옵션	341
git apply 명령	337
git archive 명령	365

git bisect 명령	100–106, 126, 182	git stash show	225
git bisect replay 명령	104	HEAD^ HEAD	367
git bisect reset 명령	106	--ours	162
git bisect visualize 명령	105	--staged	435
git blame 명령	106, 182	--stat 옵션	90
git branch 명령	114, 115, 127–129	--theirs	162
-a 옵션	259	경로 제한	144–147
--track	280	대화식 링크 스테이징	431
git cat-file 명령	166, 427	예제	137–141
-p 옵션	56	옵션	136
git checkout 명령	119–127, 151, 166, 201, 331	일반적인 형태	140
-m 옵션	169	충돌 해결	160–163
날짜 기반 체크아웃	421–422–425	커밋 범위	141–144
병합	149	파일 버전 비교	166
커밋의 파일 버전	427	git diff-tree 명령	462
git check-ref-format 명령	112	git fetch 명령	84, 262–263, 243, 271
git cherry-pick 명령	198–200, 400, 460	refspec	249
git clean 명령	462–463	후크 스크립트	349
git clone 명령	32, 252, 266–268, 370	git filter-branch 명령	385, 411–421
--bare	241–242	--all 옵션	454
후크	347	--index-filter	451
후크 스크립트	349	--msg-filter	419, 457
git commit 명령	26, 28, 54, 57	--subdirectory-filter	454
--all 옵션	64–69	--tag-name-filter cat	454
--amend 옵션	22, 203–206	--tree-filter	451
--dry-run 옵션	225, 231	예제	414–420
--interactive	63	커밋 메시지 편집하기	419–420
--no-verify 옵션	353	파일 지우기	414–419
영향	77	이후에 수동으로 git gc 실행하기	451
저장소에 있는 파일 삭제하기	31	적용 범위	421
git commit-tree 명령	54	git format-patch 명령	309, 321, 323–334, 370
git config core.logAllRefUpdates 명령	233	git diff 명령과 비교	323
git config 명령	27, 33, 243, 276	--pretty=format	336
--global 옵션	468	--root	331
매개변수, 가비지 콜렉션	453	git fsck 명령	129, 441–446
git diff HEAD 명령	162	git gc 명령	129, 452–453
git diff MERGE_HEAD 명령	162	git hash-object 명령	62
git diff 명령	30, 58, 131, 132–137	git help hooks 명령	354–358
-a 옵션	137	git help svn 명령	394
--cached 옵션	134	git help 명령, --all 옵션	22
git format-patch 명령과 비교	323	git init 명령	24, 46

git log 명령	28, 87–90, 95, 107, 258, , 393, 407, 426	--onto 옵션	207, 303
--abbrev-commit 옵션	88, 153, 229, 397	--preserve-merges 옵션	219
--follow 옵션	70, 466	--skip 옵션	208
git format-patch 명령과 비교	323	병합과 비교	214–220
--graph 옵션	153	git reflog delete 명령	237
--left-right	163, 164, 397	git reflog expire 명령	237, 453
--merge	163, 164	git reflog show 명령	233
origin/master..master	282	git reflog 명령	129, 196
-p 옵션	89, 163, 164	git remote 명령	243, 253–256, 274–276
--pretty=oneline 옵션	153, 229, 397	add origin	275, 296
--pretty=short 옵션	88	prune	275
--stat 옵션	90	rename	275
중복된 항목	217	rm	275
총들 해결	163–164	show origin	275
git ls-files 명령	58, 62, 166	update	255, 275
-u 옵션	159	git rerere 명령	469
git ls-remote origin 명령	257	git reset 명령	188–197, 201
git ls-remote 명령	244, 249	--hard ORIG_HEAD	272
git ls-tree 명령	386	--hard	169, 189, 192, 195, 197, 230
git merge 명령	149, 401	--mixed	189
--no-fl 옵션	406	--soft	188, 192, 193, 197
git merge 명령 다시 하기	235	손실된 커밋	441
git merge-base 명령	95, 113	최상위 커밋 제거	190–191
git mv 명령	31, 69	git revert 명령	200, 202
git pull 명령	243, 249, 261, 271, 309	git rev-list 명령	422, 425
--rebase 옵션	227, 460	--no-merges	330
-s ours	366	-v	330
-s subtree	362, 367	git rev-parse 명령	86, 236, 387
-s subtree 하위 프로젝트 가져오기	364–369	git rm 명령	31, 57, 66–69
병합 또는 리베이스 옵션	263–266, 265	--cached	67
수정 사항 충돌로 인한 실패	226	git send-email 명령	322–334
하위 프로젝트 업데이트	371	git show 명령	29, 168
git push 명령		--pretty=fuller	55
228, 244, 249, 256–257, 260, 261, 309, 356, 475		git show-branch 명령	
--mirror origin	297	29, 86, 116–118, 127, 231, 256, 339	
-u 옵션	296	--more 옵션	117
베어 저장소	284–285	git show-ref 명령	249
git rebase 명령	206–220, 400, 449, 460	git stash branch 명령	229
--abort 옵션	209	git stash drop 명령	224
--continue 옵션	208	git stash list 명령	224–225
-i(--interactive) 옵션	209–214	git stash pop 명령	223, 228

git stash save 명령	222	git 데몬, 게시하기	295
git stash show 명령	225	Git 설치 패키지	9
git stash 명령	221–232	최신 버전 확인하기	12
--all 옵션	226	Git 설치하기	9
--include-untracked 옵션	226	Windows	15–19
어지러운 작업 디렉터리	449	리눅스 바이너리 배포판	9–11
git status 명령	25, 58, 59, 62, 64, 159, 190, 282	최신 버전 패키지 확인하기	12
git submodule add URL 명령	387	Git 저장소	
git submodule add 명령	377, 382	pu(제한된 업데이트) 브랜치	304
git submodule foreach 명령	383	병렬로 SVN 저장소 관리하기	456
git submodule init 명령	377, 383	Git 전송 프로토콜	321
git submodule status 명령	382	Git 태그	498
git submodule summary 명령	383	Git 호스팅 서비스	297
git submodule update 명령	126, 378, 383	git/.git 디렉터리, gitlink	374
git submodule 명령	372–379	GIT_AUTHOR_EMAIL 환경 변수	27
git svn clone 명령	391–395	GIT_AUTHOR_NAME 환경 변수	27
--prefix=svn/	402	GIT_EDITOR 환경 변수	26
--stdlayout	402	git-core 패키지	9
git svn create-ignore 명령	408	git-daemon 명령	246, 291–292
git svn dcommit 명령	401, 407, 409	--export-all	291
git svn rebase 명령		git-daemon-run	10
git svn 명령과 비교	400	git-email 명령	10
커밋하기	398	git-grep 명령	
git svn 명령	391, 406	명령행 옵션	464
--authors-file	401	저장소 검색하기	463–465
git svn rebase 명령과 비교	400	git-gui	10
작업 디렉터리	393	git-http-backend 명령	294
캐시 재생성하기	408–409	GitHub	
푸시, 풀, 분기 및 병합	399	Advanced Search 페이지	490
git svn 저장소, 모든 브랜치 복제하기	402–404	Explore 페이지	490
git symbolic-ref 명령	84	Git 커밋을 위한 자동 병합 단추	486
git tag 명령	55	Mozilla의 Ace 컨트롤	495
Git Tower	389	New Repository 페이지	475
git update-index 명령,		계정 만들기	472
--assume-unchanged Makefile	469	네트워크 그래프	481
git update-ref 명령	466	뉴스 피드	479
Git URL	246	와처	478
git whatchanged 명령	461–462	위키	491
--since= 옵션	461	저장소 만들기	296, 475
git write-tree 명령	49, 54	저장소에 로컬 콘텐츠 넣기	475
Git 네이티브 프로토콜	247, 322	저장소에 있는 하위 모듈 하이퍼링크	390

- | | |
|-------------------------------|---------|
| 조직 | 499 |
| 통지 | 487 |
| 페이지(웹 사이트용 Git) | 492 |
| 풀 요청 | |
| 관리하기 | 484–487 |
| 생성하기 | 482–484 |
| 프로젝트 아카이브 만들기 | 498 |
| 프로젝트 포크하기 | 319 |
| 프로젝트의 개인용 포크 | 480–482 |
| 하위 모듈의 하이퍼링크 | 390 |
| GitHub 개발을 위한 부분 오픈 소스 모델 | 505 |
| GitHub 개발을 위한 중재자 모델 | 504 |
| GitHub 계정, 만들기 | 472 |
| GitHub 엔터프라이즈 | 506 |
| GitHub 저장소에 대한 README 넣기 | 475 |
| GitHub.com | 317 |
| GitHub에 대한 커뮤니티의 기여, 투명성과 보편성 | 481 |
| GitHub의 개인 계정 | 473 |
| gitk(Git 브라우저) | 10, 105 |
| 커밋 그래프를 보기 위한 | 94–95 |
| gitlink | 373–376 |
| git add 명령으로 업데이트하기 | 379 |
| Gitolite 프로젝트 | 290 |
| Git-over-SSH 프로토콜 | 297 |
| git-submodule.sh 셸 스크립트 | 376–379 |
| git-svn-id 명령 | 407 |
| git-svn-id, 커밋 메시지 | 395 |
| gitweb | 10 |
| gitweb 프로그램 | 179 |
| Git의 구성 파일 | 33–35 |
| 변수 설정 나열하기 | 34 |
| Git의 옵션, 나열하기 | 21 |
| Git의 하위 명령, 나열하기 | 21 |
| --global 옵션 | 33 |
| git config 명령 | 469 |
| --graph 옵션, git log 명령 | 153 |
| grep 명령(Unix) | 352 |
-
- | | |
|-------------------------|--------------|
| H | |
| --hard 옵션, git reset 명령 | 169 |
| HEAD 참조 | 84 |
| git diff 명령에서 간접적으로 참조 | 134 |
| git reset | 201 |
| 독립된 | 102, 125–127 |
| 조정하기 | 188 |
| HTTP 데몬, 저장소 게시하기 | 293, 295 |
| HTTP 프로토콜 | 247, 322 |
| HTTPS 프로토콜 | 247 |
-
- | | |
|--------------------------------------|---------|
| I | |
| -i(--interactive) 옵션, git rebase 명령 | 209–214 |
| --include-untracked 옵션, git stash 명령 | 226 |
| index-filter | 412 |
| inetd 서비스, git-daemon | 291 |
| INSTALL 파일, 외부 종속 항목 | 13 |
| --interactive 옵션 | |
| git add 명령 | 63 |
| git commit 명령 | 63 |
-
- | | |
|-------------------------------------|-----|
| J | |
| JavaServer Pages(JSP) | 493 |
| Jekyll | 493 |
| JSON(JavaScript Object Notation) 형식 | 500 |
-
- | | |
|---------------------------------|-----|
| K | |
| KDE(K Desktop Environment) 프로젝트 | 361 |
-
- | | |
|---|----------|
| L | |
| LDAP(Lightweight Directory Access Protocol) | 506 |
| --left-right 옵션, git log 명령 | 163, 164 |
| libcurl 라이브러리 | 13 |

Lightweight Directory Access Protocol(LDAP)	506
Linux Foundation Publications	300
LocationMatch 지시자, 익명 읽기 액세스	294
log	git log 명령 참조
ls-files	git ls-files 명령 참조
ls-remote	git ls-remote 명령 참조

M

-m 옵션, git checkout 명령	169
--M 옵션, git diff 명령	136
M. J. 록킨드(M. J. Rochkind)	5
make 명령, Git 설치	13-14
Makefile	467-468
NO_EXPAT 옵션	13
mbx 스타일 메일 폴더, git format-patch 명령으로 만들기	335
MERGE_HEAD	84
--mirror origin 옵션, git push 명령	297
mod_alias 모듈(Apache)	294
mod_cgi 모듈(Apache)	294
mod_env 모듈(Apache)	294
--more 옵션, git show-branch 명령	117
Mozilla의 Ace 컨트롤	495
msg-filter	413
msgfmt 유틸리티	13
msysGit	15
설치하기	17-19
MTA(mail transfer agents)	336
MUA(mail user agent)	334, 336
mutt, 메일박스 가져오기	335

N

netdev 저장소	316
Network File System(NFS)	246
--no-ff 옵션, git merge 명령	406
--no-verify 옵션, git commit 명령	353

O

Objective-C 에코시스템, 하위 모듈	383
Octopress	493
oh-my-zsh	388
--onto 옵션, git rebase 명령	207, 303
openssl 라이브러리	13
ORIG_HEAD	84
origin/HEAD 참조	259
ours 병합 전략	177
--ours 옵션, git diff 명령	162

P

-p 옵션	
git add 명령	429, 432
git cat-file 명령	56
git log 명령	89, 163, 164
parent-filter	413
patch 명령	337
PHP	493
pickaxe	107
post-applypatch 후크	355
post-checkout 후크	349, 358
post-commit 후크	354
post-merge 후크	358
post-receive 후크	357
post-update 스크립트	293
post-update 후크	349, 357
PowerPC 아키텍처	314
pre-applypatch 후크	355
pre-auto-gc 후크	358
pre-commit 후크	349, 353
만들기	353
prepare-commit-msg 후크	354
pre-rebase 명령	358
pre-receive 후크	357
--preserve-merges 옵션, git rebase 명령	219
--pretty=oneline 옵션, git log 명령	153, 229, 393
--pretty=short 옵션, git log 명령	88

R

- r 옵션, Unix diff 132
- RCS(Revision Control System) 6
- rebase 옵션, git pull 명령 227, 459
- rebase-apply 디렉터리 341
- recurse-submodules 옵션 389
- git 387
- reflog 232-237, 507, git reflog 명령 참조
 - 만료 관련 기본 제한 시간 조정 455
 - 제거하기 444
 - 항목 제거 453
- refs/remotes/ 네임스페이스 245
- refs/stash 참조 223
- refspec 245, 248-250
 - 푸시 282
- remote git remote 명령 참조
- rerere 기능 468
- reset git reset 명령 참조
- REST API 500-501
- rev-list git rev-list 명령 참조
- RFC 1738 246
- RFC 2396 246
- Rsync 프로토콜 248

S

- SCCS(Source Code Control System) 5
- Secure Shell(SSH) 명령, msysGit 17
- sed 명령 419-420
- SHA1(Secure Hash Function) 3, 40
 - 인덱스 비교하기 79
 - 해시 계산하기 및 인쇄하기 62
 - Git에서의 사용 50-52
 - 번호 줄이기 82
 - 해시의 고유성 48
- show-branch git show-branch 명령 참조
- Simple Mail Transfer Protocol(SMTP) 322, 335
- since= 옵션, git whatchanged 명령 461
- skip 옵션, git rebase 명령 208

- Smart HTTP, 저장소 게시하기 293-295
- SMTP(Simple Mail Transfer Protocol) 322, 335
- SparkleShare 471
- SSH(Secure Shell)
 - Git 네이티브 프로토콜 터널링 247
 - mysysGit의 명령 17
 - 저장소 액세스 290
- stash 221-232, 507, git stash 명령 참조
 - 상태를 재생성하기 231
 - 상태에 저장된 컨텍스트 다시 만들기 224-225
- stash 저장, 로그 메시지 222
- stat 옵션 137
- git diff 명령 137, 144, 145
- git log 명령 90
- git stash show 명령 225
- subdirectory-filter 413
- sudo 명령 XXI
- svn git svn 명령 참조 71
- svn mv 명령 71
- SVN(Subversion)
 - Git과의 상호 운용성 15
 - Git에서 소스 코드 변경하기 395-396
 - svn:ignore와 .gitignore 비교 408
 - 다시 병합하기 406-408
 - 두 저장소의 브랜치 조작하기 458
 - 리비전 정보 24
 - 리비전 추적 147
 - 변환 팁 455-458
 - SVN 가져오기 후 트렁크 제거하기 456
 - SVN 커밋 ID 제거하기 457
 - 고유한 Git 명령 만들기 460
 - 저장소 복제본 391
 - 커밋 전에 가져오기 396-398
 - 파일 이름 변경 추적하기 71
- svn:ignore, .gitignore 파일과 비교 408
- system 옵션 33

T

- tag-name-filter 413

Teams, GitHub	499
--theirs 옵션, git diff 명령	162
tree-filter	412

U

-u 옵션	
git ls-files 명령	159
git push 명령	296
unified diff	131
Uniform Resource Locators(URL), Git 지원	246
Unix diff	132
--unset 옵션	35
update 후크	349, 357
Updating origin 메시지	255

V

VCS의 데이터 무결성	3
VCS의 무결성	3
VCS의 원자성 트랜잭션	4
--version 옵션, git 명령	22

W

Windows 환경	34
Windows, Git 설치하기	15-19
WIP(work in progress)	222
캡처를 위한 stash	221

X

X.org 프로젝트	302
xinetd 서비스	291

Y

yum 명령(페도라)	11
-------------	----

Z

ZIP 파일, 가져오기	364
zlib 라이브러리	13

ㄱ

가비지 컬렉션	237, 451-453
커밋과 블록	129
가져오기, 커밋 전	396-398
간단한 복제본, 단일 SVN 브랜치	391-395
간편 태그 유형	55
개발	
분산	303-305
업스트림 리베이스에서 복구하기	458-460
프로젝트에서의 위치	306-310
개발 히스토리 조작하기	186
개발 브랜치	111, 244
개발 저장소	240
저장소의 복제본	258
푸시	285
개발 프로젝트, 분산 환경 히스토리	304
개발자	307
관리자와의 상호 작용	308
권한이 있는 저장소에 새 개발자 추가하기	258-260
객체 ID	40
객체 모델	74-76
병합	180
객체 저장소	69
그림	44
인덱스 스냅샷	79
파일 콘텐츠 항목	48
객체, 도달할 수 없는	451

검색하기	
저장소, git-grep 명령	463-465
커밋의 파일 버전	427
게이트키퍼 저장소	402, 404-405
결합된 diff 형식	156
병합 커밋	167
경로 이름과 콘텐츠	42
경로 제한, git diff 명령	144-147
계층 구조	
여러 .gitignore 디렉터리	73
트리 객체	52-53
공용 코드, 저장소	472-474
공용 히스토리, 변경하기	303
공유 라이브러리	360
공유 저장소 구조	298
관리자(maintainer)	299, 300, 307
개발자와의 상호 작용	308
교호적인 현실주의적 히스토리	186
구성 파일의 변수, 모든 변수의 설정 나열하기	34
군사체계식 GitHub 개발 모델	504
권한	251
권한이 있는 저장소	
만들기	251-256
새 개발자 추가하기	258-260
그래들 멀티프로젝트 빌드	383
그래프	91
도달 가능성	96
위상 정렬	346
기본 메시지, pre-commit-msg 후크	354
기본 브랜치, 저장소	111
기본 포트	
Git	247
inetd 서비스	291
깨끗한 작업 디렉터리	26

L

나쁜 패치	346
날짜 기반 체크아웃	422-425
내부 설계, VCS	5

네트워크 액세스, 제공	290
논패스트포워드 푸시	269-270
뉴스 피드, GitHub	479
느낌표(!), 역 패턴	73

C

다시 연결하기, 손실된 커밋	446
다운스트림 생산자/게시자	310
다운스트림 소비자	310
다운스트림 저장소, 커밋 전송 메커니즘	309-310
다운스트림 플로	306
다이제스트	51
대괄호([]), git show-branch 출력	116
대칭 차집합	99
대화식 리베이스, 어지러운 작업 디렉터리	449-450
대화식 형크 스테이징	428-441
더티 트리로 가져오기	226, git pull 명령 참조
더하기 기호(+)	
diff	131, 162
git show-branch 출력	116
refspec	248
분리 작업에서의 위치	437
커밋 차이점을 볼 때	30
더하기 기호(+++), diff	131
데비안(Debian) 환경	XVIII
Git 설치하기	9-10
데이터 객체, 불변성	4
델타, 파일 간	43
도달 가능성	
가비지 컬렉션	451
그래프	96
도달할 수 없는 객체	441, 451
도움말	22, 507
git help 명령	git help 명령 참조
문서	문서 참조
GitHub	473
독립된 HEAD 브랜치	102, 125-127, 280
동료 검토, “패치, 이메일, 검토, 적용” 패러다임	323
동시 버전 시스템(CVS)	6

리비전 정보	24
동적 커밋 히스토리	304
디렉터리	
Git 리비전 정보	24
구조 정리	462–463
슬래시 문자(/), 이름의 후미	72
현재 브랜치를 새롭게 선택했을 때의 영향	120
디팟 저장소	251, 288
변경 사항 푸시하기	260–266
딕 그룬(Dick Grune)	6

리

라이브러리, 공유	360
래리 월(Larry Wall)	337
로그 메시지	
Git 커밋	26
stash 저장	222
로컬 저장소	243
원격 추적 브랜치	243
로컬 추적 브랜치	240, 244
만들기	280
원격과 비교하기	281
이름	279
루트 권한, 필요한 명령	XXI
리눅스 바이너리 배포판, Git 설치하기	9
리눅스 커널 프로젝트, 저장소 예제	300–302
리베이스	183, 507
git rebase 명령	git rebase 명령 참조
브랜치, 후크	358
커밋	206–220
리베이스 충돌, 솔루션 프로세스 자동화하기	468
리비전 관리 시스템(RCS, Revision Control System)	
	6

리

마일스 조지(Miles Georgi), externals 스크립트	372
머큐리얼(Mercurial)	5
메이븐 멀티모듈 프로젝트	383

메타데이터, SVN 변환	456
명령 별칭, 설정하기	35
명령, 고유한 명령 만들기	460
모노톤(Monotone)	7
모듈화, 하위 모듈	383
무시된 파일	59, 72–74
예외	74
문서	
git 하위 명령어	22
git-grep 명령	464
온라인 Git 문서	23
전체 빌드	14
물결 표시(~)	84–86
물리적 데이터 레이아웃, Git	42

비

바트 마시(Bart Massey)	467
방향성이 있는 비순환 그래프(DAG)	91
백업	
인터넷	306
전략	1
피어투피어	306
백업을 위한 인터넷	306
백워드 포트	207
버그	
수정할 브랜치 지정하기	115
커밋 분리하기	100
버전 관리 시스템(VCS)	1
분산	149
원하는 기능	2–5
전례	5–7
번호 기호(#)	
.gitmore 파일	72
주석	157
베어 저장소	240, 292
git push	284–285
만들기	252–253
벤 헤렌스미트(Ben Herrenschmidt)	314
변경 로그	4

변경 사항, 빠른 개요	461-462	커밋 그래프	93, 94
별칭 구성하기	35	병합 표시줄	159
별칭, 구성하기	35	병합된 히스토리, 원격 저장소에 넣기	273
별표(*)		보기	
git show-branch 출력	116	기존 커밋	87-90
글로벌	72	병합 커밋	168
브랜치 이름의 규칙	111	브랜치	116-118
병합	113, 149-183, 507	커밋 차이점	32
criss-cross	172, 176, 179	커밋	28-30
dcommit	407	복구하기	
git merge 명령	git merge 명령 참조	삭제된 브랜치	129
git rebase 명령과 비교	214-220	손실된 커밋	441-446
Git 객체 모델	180	복사하기, 하위 프로젝트 가져오기	364
변경 사항을 다른 브랜치로	122-124	복제된 하위 모듈, 읽기 전용 주소	385-386
브랜치	4, 150-153	복제본, 간단한, 단일 SVN 브랜치	391-395
브랜치 리베이스하기	216	부모 저장소	306
솔루션 프로세스 자동화하기	468	부분 체크아웃	360-362
예제	150-158	분리하기, 저장소	453-454
전략	170-173	분산 개발	303-305
다중 브랜치	170-173	VCS	2
일반 병합	175-177	업스트림 리베이스에서 복구하기	458-460
적용하기	177-179	프로젝트에서의 위치	306-310
퇴화 병합	173-175	프로젝트에서의 위치	
준비하기	150	업스트림 및 다운스트림 플로	306-310, 307
취소 또는 다시 시작하기	169	분산 버전 관리 시스템(DVCS)	149
특수	177	분산 저장소 구조	299-300
패치와 비교	346	분할 개발, VCS	4
병합 다시 시작하기	169	불변성, 데이터 객체	4
병합 드라이버	179	브랜치	109-129, 149, 507
병합 충돌	153-158	git branch 명령	git branch 명령 참조
쓰리웨이 병합으로 해결하기	343	다른 브랜치로 변경하기	201
원격 저장소 개발	272	독립된 HEAD	125-127
조사를 위한 git diff	156	두 저장소의 브랜치 조작하기	458
해결	158-169	만들기	114-116
마무리하기	167-169	변경 사항에 대한 기록	232
충돌 조사하기	160-165	변경 사항을 다른 브랜치로 병합하기	122-124
충돌이 발생한 파일 찾기	159	병합	150-153, 병합 참조
병합 충돌 표시	343	병합과 함께 리베이스하기	216
병합 커밋		보기	116-118
변경된 파일이 없음을 확인하기	367	사용하기	112-114
보기	168	사용하는 이유	109-110

삭제하기	127-129	상대적 이름, 커밋 식별	84-86
새 브랜치를 만들고 체크아웃하기	124-127	상세 태그 유형	55
원격 저장소		상용 Git 호스팅 서비스	297
삭제된	275	상위 커밋	85, 168
이름 변경	283	서버	287-288
정보 보기	257	Git 명령	393
원격 추적	240	서브버전 브리지	496
브랜치 목록에 포함시키기	116	서브버전 저장소, Git과 함께 사용	391-409
이름	111-112	서브트리 전략	177
이름 나열하기	115	성능	
읽기 전용	459	VCS	2
저장소에 있는 모든 브랜치 나열하기	259	후크 영향	348
체크아웃하기	119-127	소셜 코딩	471
추적하기	244-245	오픈 소스	477
커밋 존재	116	폐쇄적 소스	502
태그 객체와 비교	110	소스 릴리스	12
히스토리, 보기	87	소스 코드 관리자(SCM)	1
브랜치 게시하기	113	버전 관리 시스템	버전 관리 시스템(VCS) 참조
브랜치의 팁	113	손실된 데이터, git fsck로 위치 찾기	441-446
변경 사항에 대한 기록	232	손실된 커밋, 다시 연결하기	446
브랜치의 헤드	113	수많은 개발자를 관리할 수 있는 확장성	3
비교	131	수정된(dirty) 상태	
diff 및 git diff 명령	diff 및 git diff 명령 참조	인덱스	77
객체, 해시 함수	51	작업 디렉토리	150
비추적(nontracking) 브랜치	244	대화식 리베이스	449-450
빈 디렉터리 만들기	358	재설정	169
빌드, 시작	12	트리 객체	226
빼기 기호(-)		숨겨진 파일	47
diff	131, 162	셸 글로벌 문자	72
git show-branch 출력	116	셸 환경 설정, 만들기 또는 변경하기	412
분리 작업에서의 위치	437	스쿼시 병합	182-183
빼기 기호(---), diff	131	스쿼시 커밋	182
		스크립트	
		고유한	460
		하위 프로젝트 체크아웃하기	370-372
		스테이징	
		대화식 힙크	428-441
		인덱스의 변경 사항	40
		파일	24
		git add 명령	61-69
		git diff 명령	136
사용자 식별, SVN과 Git 비교	394		
삭제하기			
브랜치	127-129		
저장소의 파일	31		
참조	465		

슬래시 문자(/), 후미, 디렉터리 이름 72, 111, 374
 시간 소인
 커밋 305
 커밋 그래프 92
 시그윈 기반 Git 패키지 설치하기 15
 신뢰성, VCS 3
 심볼릭 이름, 커밋 234
 심볼릭 참조(symref) 83-84
 쓰리웨이 병합 175, 342-345

○

암호, 원격 호스팅 시스템 290
 압축 파일, 가져오기 364
 애플리케이션, 가시성, 제한으로서의 하위 모듈 389
 액세스 권한 제한, 저장소 게시하기 289-290
 액세스 제어 레벨, 선택 296
 액세스 제어 목록(ACL, Access Control List) 358
 업데이트를 저장소로, 가져오기 260-266
 업스트림 리베이스, 복구 458-460
 업스트림 브랜치, 이름 변경 459
 업스트림 생산자/게시자 310
 업스트림 소비자 309
 업스트림 저장소 243
 다른 저장소로 변환하기 312-314
 여러 저장소 사용하기 314-316
 커밋 전송 메커니즘 309-310
 하위 프로젝트 변경 사항 제출하기 370
 업스트림 플로 306
 역할 이중성 309-310
 오류 메시지 26
 “프리” 스크립트에서 인쇄하기 353
 “branch ‘bug/pr-3’ is not an ancestor of your current HEAD” 127
 “failed to push some refs” 270
 “not found: did you run git update-serverinfo” 293
 “Perhaps git-update-server-info needs to be run there?” 293
 “Your local changes to the following files will be overwritten by checkout” 120

오른쪽(>), git log --left-right 표시 164
 오픈 소스 라이브러리, 찾기 및 사용하기 490
 오픈 소스, 소셜 코딩 477
 오픈 소싱, 최종 단계 502
 옥토퍼스 병합 전략 175
 옮겨진 파일 따라가기 466-467
 와치, GitHub 478
 외부 종속 항목, INSTALL 파일 13
 왼쪽(<), git log --left-right 표시 164
 우분투(Ubuntu) 리눅스 환경
 Git 설치하기 9-10
 운영 체제 XVIII
 원격 239, 243-244
 원격 저장소 239-285
 개발 주기 266-274
 대체 히스토리 가져오기 270
 대체 히스토리 268
 병합 충돌 272
 병합된 히스토리 넣기 273-274
 비패스트 포워드 푸시 269-270
 저장소 복제하기 266-268
 히스토리 병합하기 271-272
 구성 274-276
 git config 사용하기 276
 git remote 명령 274-276
 수동 편집 277
 다중 277
 변경 사항 푸시하기 256-258
 브랜치 정보 보기 257
 브랜치 추가 및 삭제하기 282-283
 사용 예제 251-266
 삭제된 브랜치 275
 참조 나열하기 249
 현재 저장소와의 링크 254
 원격 추적 브랜치 240, 242, 244, 257, 277
 로컬과 비교하기 281
 브랜치 목록에 포함시키기 116
 원본 리모트 253-256
 원본, 복제본 243
 원자적 변경 세트 80
 월터 티치(Walter Tichy) 6

날짜 기반 한정자	235
변경 사항에 대한 기록	232
업데이트 및 삭제하기	465
파악을 위한 도구	236
참조 업데이트하기	465
참조되지 않은 객체	441
참조되지 않은 커밋	451
찾기	
오픈 소스 라이브러리	490
커밋	100–107
책임성, VCS 강제성	4
체크아웃하기	507
git checkout 명령	git checkout 명령 참조
부분	360–362
브랜치	119–127
커밋되지 않은 변경 사항	120–122
하위 프로젝트, 사용자 정의 스크립트	370–372
최상위 디렉터리, SVN 가져오기 후 제거하기	456
최상위 커밋	
변경하기	203–206
제거하기 위한 git reset	190–191
최종 단계에서의 오픈 소싱	502
추적 브랜치	110, 244–245, 277–281
추적되지 않은 파일	59
추적된 파일	59
git add 명령	62
추적하기, 파일 이름 변경	71–72
축약, 캐럿(^)과 물결 표시(~)	85
충돌	51, 병합 충돌 참조
SHA1	48
충돌 조사하기	160–165
충돌이 발생한 파일 찾기	159
취소(aborting)	
리베이스	209
병합	169, 272
커밋 로그 메시지	66
커밋, pre-commit 후크	353
커밋되지 않은 변경 사항	120

ㅋ

캐럿(^)	84–86
브랜치 이름	117
브랜치 이름 규칙	111
캐시, git svn용 재생성하기	408–409
커널 개발 주기	301
커밋	44, 79–107
2단계 프로세스	57
게시와의 분리	304
리베이스하기	206–220
만들기	53–55
메타 정보, 편집하기	205
버그가 있는 커밋 분리하기	101
보기	28–30
복구를 위한 팁	454
선형화	333
세부사항 보기	55
손실된 커밋 다시 연결하기	446
손실된 커밋 복구하기	441–446
수정하기	185–220
이유	185
주의 사항	187–188
최근	203–206
식별하기	81–87
상대적 이름	84–86
절대적 이름	82–84
실행 취소 효과	200
심볼릭 이름	234
차이점	134
차이점 보기	30
참조되지 않은	451
찾기	100–107
커밋 전송 메커니즘	309–310
하위 모듈의 해시	386
히스토리	87–100
동적	304
커밋 ID	81–87, 400–402
git svn	395
SVN 제거하기	457

커밋 객체	39
링크	373
커밋 관련 후크	354–355
커밋 권한	288
커밋 그래프	90–95
보기 위한 gitk	94–95
히스토리	87
커밋 로그 메시지	66
편집기 설정	35
커밋 메시지	
git-svn-id	395
편집하기	413
git filter-branch	419–420
커밋 범위	95–100
git diff 명령	141–144
git format-patch	329
범위 해석	329–330
커밋 수정하기	185–220
이유	185
주의 사항	187–188
최근	203–206
커밋 식별하기	81–87
상대적 이름	84–86
절대적 이름	82–84
커밋 작성자, 구성하기	26–27
커밋 히스토리	
재작성	411
저장소 분리 시 보존하기	453
커밋되지 않은 변경 사항, 체크아웃하기	120–122
커밋의 메타 정보, 편집하기	205
커밋의 선형화	333
커밋의 역순, 적용하기	200
케이스 패커드(Keith Packard)	302
코드 샘플을 오픈 소스로 제공하기	388
코딩 모델	503–505
콘텐츠 주소화 이름	40
콘텐츠 주소화 파일 저장소	7
콘텐츠 추적 시스템, Git	40
콘텐츠와 경로 이름	42
콜론(:)	
refspec	248

브랜치 이름 규칙	111
-----------	-----

E

태그 객체	39, 55
브랜치와 비교	110
추적 카운트를 위한 이름	113
텍스트 병합 드라이버	180
템플릿 후크	350
토발즈, 리눅스	58, 300, 306, 359, 373
포크하기	319
토픽 브랜치	110, 244
통계, 차이점	137
통지, GitHub	487
통합 브랜치	110
퇴화 병합	173–175
트리 객체	39, 44, 49
git diff 명령으로 비교	134
계층 구조	52–53
링크	373
저장소를 나타내는	91
현재 인덱스에서 만들기	49
특수 병합	177

F

파운드 기호(#)	
.gitmore 파일	72
주석	157
파일	
개별 수정 라인	106
경로 이름과 콘텐츠	42
구버전 가져오기	425–427
다른 버전 비교하기	166–167, 166
모든 버전의 저장소	41
숨겨진	47
옳겨진 파일 따라가기	466–467
이동 또는 이름 변경	69–70
이름 변경 추적하기	71–72

저장소에서 제거하기	31-32	병합을 위한 템플릿 메시지	157
준비 전 상태로 되돌리기	190	원격 저장소 구성	276
준비된 상태에서 준비되지 않은 상태로 변환	67	커밋 로그 메시지	35, 66
지우기를 위한 git filter-branch 명령	414-419	편집기 파일, 필요 없는 파일 제거하기	451
파일 간 차이점	43	편집하기	
필요 없는 편집기 파일 제거하기	451	커밋 메시지	413
히스토리	70	git filter-branch	419-420
파일 분류	59-61	커밋의 메타 정보	205
파일 이동하기	69	폐쇄적 소스, 소셜 코딩	502
파일 이름	48	폐쇄적 저장소, GitHub	296
명시적으로 식별하기 위한 이중 대시	23	포스트 후크	348
무시된 파일의 패턴	72-74	포워드 포팅	207
변경하기	31	포트, 기본	
파일 이름 변경	31, 69	Git	247
추적하기	71-72	inetd 서비스	291
파일 저장소, 콘텐츠 주소화	7	포함과 첨부 비교, 패치	336
파일 지우기, git filter-branch 명령	414-419	푸시 관련 후크	356-357
파일을 이전 상태로 되돌리기	190	푸시 작업	
파일의 구버전 가져오기	425-427	로컬 콘텐츠를 GitHub에	475
파일의 버전, 비교하기	166-167, 166	비패스트 포워드	269-270
패스트포워드 병합	263, 270	풀 요청	319
패스트포워드(fast-forward) 퇴화 시나리오	174, 177	GitHub	
패치	321-346	관리하기	484-487
교환을 지원하는 명령	322	만들기	482-484
나쁜	346	프랑수아 마리 아루에(Fran?ois-Marie Arouet)	411
메일 보내기	334-337	프레드릭 퀴비넨(Fredrik Kuivinen)	178
병합과 비교	346	프로젝트	
사용 이유	322-323	GitHub에서 아카이브 만들기	498
생성하기	323-334	결합하기	359-379
위상 정렬	333-334	지침	428
적용하기	337-345	코드 가져오기	362-372
지침	428	포크하기	317-319
커밋 세트 확인하기	330	프로젝트 결합하기	359-379
패치 관련 후크	355-356	프로젝트 포크하기	317-319
패치 이메일 보내기	334-337	GitHub	319
패치 첨부와 포함 비교	336	개인용, GitHub	480-482
패턴, .gitignore와 파일 이름	72	조정	318
팩 파일	39, 43	프로젝트로 코드 가져오기	362-372
페도라(Fedora)	11	프로젝트의 아카이브, GitHub에서 만들기	498
편집기		프리 후크	347
git commit 동안 열기	26	플러그인, 하위 모듈	388

피어 저장소	307	협업	288
피어투피어 백업	306	환경 변수	
피어투피어 저장소 모델	6, 284, 299, 313, 321	GIT_AUTHOR_EMAIL	27
필터링 프로세스	412	GIT_EDITOR	26
		활성 브랜치	112
		후크	347–358
		사용하는 이유	349
		설치하기	349–352
		예제	350–351
		첫 번째 만들기	351–353
		커밋 관련	354–355
		특징	348
		패치 관련	355–356
		푸시 관련	356–357
		후크 설치하기	349–355
		히스토리	
		브랜치, 보기	87
		원격 저장소의 대체 히스토리 가져오기	270
		커밋	87–100
		표시하기	28
ㅎ			
하위 노드, 커밋 그래프	92		
하위 모듈	359, 507		
git submodule 명령	git submodule 명령 참조		
명령	382–383		
모듈화	383		
모범 사례	381–390		
유스 케이스	388–389		
읽기 전용 주소로 복제된	385–386		
자격 증명 재사용	387		
준비	384		
지원	389		
커밋의 해시	386		
하위 폴더, 하위 모듈로 변환 및 추출	384		
하위 프로젝트			
가져오기			
git pull -s subtree	364–369		
복사	364		
업스트림에 변경 사항 제출하기	370		
체크아웃하기, 사용자 정의 스크립트	370–372		
하위 프로젝트 가져오기			
git pull -s subtree	364–369		
복사	364		
해결 병합 전략	175, 177		
해시 ID	82		
해시 지문	7		
형크 분리하기	432		
현재 저장소	243		