SYMBOLS
~ (tilde) as home directory alias 26

B
bare repo
  files available on 74
  remote repository (origin) as 74
bash shell 23 *see also* command line
branches
  asterisk (*) as marker for current 51
  checking out copies of 89–90
  default name for 51
  as logically separate copies 55, 55*fig*
  naming for 57, 59
  as organization for commits 7–8
  origin notation for 89
  as pointers to commits 52
  as "stack of commits" 51–52, 52*fig*, 57*fig*
  topic branches 58
  tracking 87
  as tracking method for project history 8
  as virtual copy of project 50–51
  workflow for 58–59

C
cd command 25, 26
command line
  cloning an existing project from 31–32
  Git basics for 22
  parameters (options) for 28
  prompt example 23
  starting a new project from 30–31
  using Git from 19
command reference 120–124
commit
  vs adding a file 35
  merge commit 63
  metadata available in 7
  as reference to parent commit 50
  sequential versioning of 50
commit IDs
  as data verification method 98
  as decentralized identifier 97
  hashing used to create 98
commit messages
  configuring text editor for 101
  as git log headline 99–100
  guidelines for writing 101–104
  as list of changes 104
  using active voice for 103

D

detached HEAD state
about 116
creating new branch from 117
as test environment 116
directed acyclic graph (DAG), visualizing branches with 117
. . (double dot) command 27

F

FETCH_HEAD 82, 83
file rename, overview of Git process for 43–46
file statuses
changes not staged for commit 39
changes to be committed 36
staged 36
untracked 35

G

garbage collection 36
Git
basic workflow for using 15
command line 19, 28
conceptual model of file management 16, 17
configuration options for 29
as a decentralized system 15
as managing file states 16, 17
starting a new project 30
unique identifiers for commits 15
git add command
about 35
adding new files using 40
--all option for staging files 46–48
moving files to staging area using 37
Git Bash 21
git bisect command 116
git branch command
about 51
to show all remotes on local copy 90
as way to view branches 54
git checkout command
about 28
-b option to create and switch branches 54
to switch branches 53
to view previous commits 114
git clone command
about 31
renaming local directory for 32
git command reference 120–124
git commit command
-am option to add files and commit with a message 56
example of 37
-m option to add message to 36
overview 33–34
as project snapshot 33

staging parts of files with 105
git config command, options for 29
git diff command
about 107
applications available for 109, 109*fig*
comparing commits with 109–110
GitHub output format for 108*fig*
output format for 107
git fetch command
about 88
as offline workflow tool 88
to pull remote branches 90
as snapshot of entire repo 88
git init command 30
git log command
about 40
default output for 93
options available for 94–96
to show commits 93
git merge command
about 62
example of file comparison for 64*fig*
--no-commit option for staging commit 64
git mv command, for file rename 45
Git protocol, as read-only protocol 79
git pull command
about 72
options for remotes 81
as remote merge 83
to synchronize with remote 72, 73*fig*, 79
used to reduce merge conflicts 84
git push command
about 72
creating new branch on remote 81
options for remotes 74, 80
rejection by remote 86
as remote merge 83
to set upstream remotes 87
to synchronize with remote 72, 73*fig*, 79
git remote
authentication methods for 77
networking protocols for 76–77
origin set by cloning 76
git remote add command 75–76
git remote command 76
git rm command
about 41
commit needed for 41–42
for file rename 42
vs file system removal 42
git status command
about 28, 34
file statuses 35, 36
git tag command 111

git tags 111–114
GitHub, as location for remotes 75

H
HTTPS
    as default Git hosting protocol 78
    as Git remote protocol 78

I
installing Git
    on OS X (macOS) 20
    on Windows 20

L
line endings, encoding for 21–22
ls command 26

M
macOS *see* OS X installation
merge
    to combine branches 60
    fast-forward 62, 63*fig*
    master into topic branch 65–66
    merge commit 63
    workflow for 60–61, 62*fig*
merge commit, overview of 63, 64*fig*, 65*fig*
merge conflicts
    about 65–66
    HEAD marker 68
    on remote branches 84
    steps to avoid 70, 84
    workflow to resolve 66–70
metadata, as part of a commit 7
mkdir command 24

O
objects, as Git data sources 33
origin (remote repository) 31
OS X installation 20 *see also* installing Git

P
parent commit 50
privacy concerns, sharing personal information 29–30
Pro Git book 79
project collaboration
    problems when version control is not used 10, 12*fig*
    using a "trusted system" for 10
    using distributed systems 11
    using version control for 9, 10
prompt *see* command line
push rejection, steps to resolve 86
pwd command 25

R
relative paths vs absolute paths 25–26
remote branches
  vs local copy 79
  location for 79
  naming conventions for 81
remote repository (origin)
  about 31, 71
  adding 75
  as bare repo 74
  as canonical for hub model 73, 73*fig*
  as default backup for code 73
  on GitHub 75
  hub model for 73*fig*
repository
  as file storage location 7
  initializing 31

S
sequential versioning of commits 50
shell *see* command line
SSH (secure shell)
  as Git remote protocol 77
  limitations for using 77
  protocol prefix not needed for 77
  public key authentication for 78
staging area
  about 37–38
  as local file storage only 38
  using git add for 37
system paths, symbols for 22

V
version control
  as collaboration method for projects 9, 10
  as software development practice 7
  as "trusted system" 8
  using Git as a system for 7
  using Save As as a method of 7
  as way to manage complex projects 9, 13

W
whoami command 24
Windows installation 20–21 *see also* installing Git
working directory 25