Git mastery in twenty minutes increments

Seth House <shouse@cars.com>

Cars.com

2012-07-24

Introduction

A series of twenty-minute presentations on Git.

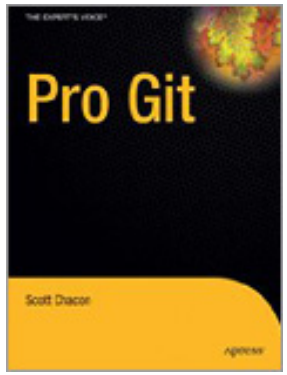


Figure: Only essential information is presented here; for in-depth knowledge check the slide notes for references and read the Pro Git book by Scott Chacon.

`http://git-scm.com/book/`

Outline

Commit objects

sha1 hash of:

- Message
- Author / committer
- Date
- Parents (if any)
- Pointer to the contents (tree)

Demonstration

- Create two commits
- View the raw commit

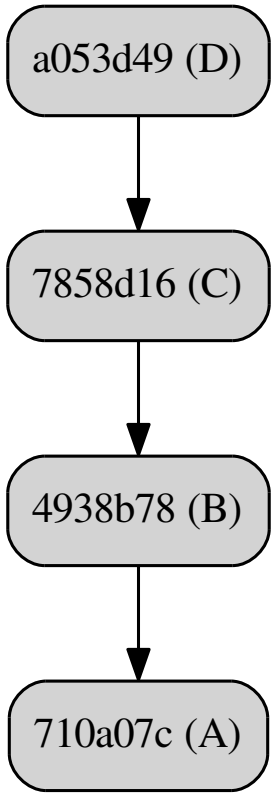
Branches

- A commit knows its parents!
- Branch names are for humans
  - master
  - origin/master

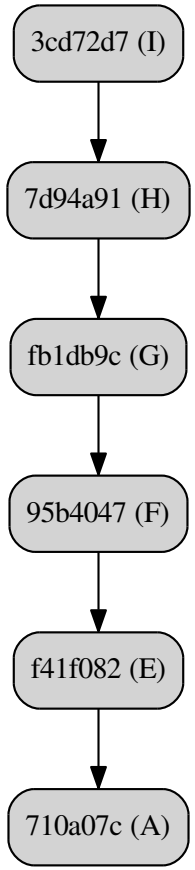
Demonstration

- Create feature branch (off master)
- Make a commit
- Merge into master
- Delete the branch

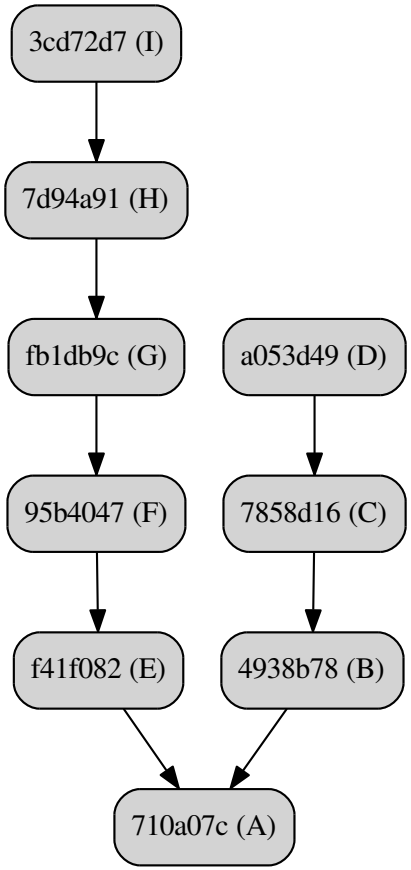
Directed acyclic graph (DAG)



Directed acyclic graph (DAG)



Directed acyclic graph (DAG)



Outline

Git refs

"refname" may refer to:

- sha1
- `master`
- `origin/master`
- `HEAD`
- `HEAD~1`
- Tags
- Many, many others; see `git-rev-parse(1)` for more

refs are for humans

- A ref points to an object
- That's it

Demonstration

- Delete the ref that points to `master`

Objects without refs are garbage collected

- Unreachable objects older than 30 days
- The reflog counts as a reference
  - reflog entries are pruned after 90 days

Demonstration

- Create a branch
- Make a new commit
- Delete that branch

Outline

Fast-forward merges

- Moves the branch pointer
- That's it

Demonstration

- Make a new branch
- Make a new commit
- Return to the original branch
- Merge the new branch

Merge commits

- A commit object with two parents

Demonstration

- Make a new branch
- Make a new commit
- Return to the original branch
- Merge the new branch using `--no-ff`

Merge conflicts

- Stages all successful automatic merges
- Surrounds conflicts with conflict markers

Demonstration

- Make a branch
- Make an edit
- Make a new branch
- Make a conflicting edit
- Merge the other branch
- Resolve the conflict
- Stage the change
- Commit (use/modify the default message)

mergetool

Three-way merge

- LOCAL
- BASE
- REMOTE
- MERGED

Demonstration

- View a merge conflict in a three-way diff program

Outline

Remotes

- Fetching from a remote adds to your local DAG
- “Remote branches” are stored locally
- You don’t need to define a remote to fetch

Demonstration

- Fetch a remote branch
- Compare commits
- Compare changes
- Merge the remote branch

Remote tracking

- At-a-glance comparison
- Syntactic sugar
- Track any ref (not just “remote branches”)