Git 1 2 3!

A fast introduction to git for professionals

Jordan Schatz **`jordan@noionlabs.com`**

You can fork this presentation at:
https://github.com/shofetim/git-1-2-3

1: What is Git?

What is Git?

Git is:

+ a ***source code management*** tool
+ distributed
+ fast

What is Git?

Linus Torvolds wrote Git to solve a problem

+ All existing version control systems where broken
+ There where about 30,000 developers working on the kernel which made other VCS painful to use
+ He needed something that made his work faster

Solution: Git

Quick Note

Why the name ’git’ ?

Quoting Linus: "I’m an egotistical ***, and I name all my projects after myself. First ’Linux’, now ’git’"

What is Git good for?

Git is a ***Source Code Management*** system, not a version control system.

What is Git good for?

Git is a ***Source Code Management*** system, not a version control system.

+ Efficient Sharing

What is Git good for?

Git is a ***Source Code Management*** system, not a version control system.

+ Efficient Sharing

+ Code Review

What is Git good for?

Git is a ***Source Code Management*** system, not a version control system.

+ Efficient Sharing
+ Code Review
+ Documentation

What is Git good for?

Git is a ***Source Code Management*** system, not a version control system.

+ Efficient Sharing
+ Code Review
+ Documentation
+ Security / Integrity

What is Git good for?

Git is a ***Source Code Management*** system, not a version control system.

+ Efficient Sharing
+ Code Review
+ Documentation
+ Security / Integrity
+ Versioning

What is Git good for?

Git is a ***Source Code Management*** system, not a version control system.

+ Efficient Sharing
+ Code Review
+ Documentation
+ Security / Integrity
+ Versioning
+ Debugging

What is Git good for?

Git is a ***Source Code Management*** system, not a version control system.

+ Efficient Sharing
+ Code Review
+ Documentation
+ Security / Integrity
+ Versioning
+ Debugging
+ A simple continuous integration server

What is Git good for?

Git is a ***Source Code Management*** system, not a version control system.

+ Efficient Sharing
+ Code Review
+ Documentation
+ Security / Integrity
+ Versioning
+ Debugging
+ A simple continuous integration server
+ more cool tools

The single most important thing you get is that:

Experimentation Is Inexpensive

Stop pussyfooting around your codebase... and start striding around like a giant

Aside

You don’t have to just use git for source code

Types of code that I use git for:

+ Ledger https://github.com/jwiegley/ledger

Types of code that I use git for:

+ Emacs Org mode http://orgmode.org/
    - Appointments
    - Notes
    - Project management
    - Time logs
    - Passwords
    - Project Proposals
    - Research papers
    - Code documentation

Types of code that I use git for:

+ Calendar
  http://www.roaringpenguin.com/products/remind

Types of code that I use git for:

+ Diagrams http://ditaa.sourceforge.net/

Types of code that I use git for:

+ Presentations
  http://docs.racket-lang.org/slideshow/index.html

Types of code that I use git for:

+ Just about everything else too

2: Get Git

Where to get it?

Git http://git-scm.com/

Linux

Mac

Windows

How it works

+ Commits

How it works

- Commits
- sha1

How it works

+ Commits
+ sha1

```
(sha1 (open-input-string "hello world"))
"2aae6c35c94fcfb415dbe95f408b9ce91ee846ed"
```

How it works

+ Commits
+ sha1

```
(sha1 (open-input-string "hello world"))
"2aae6c35c94fcfb415dbe95f408b9ce91ee846ed"
```

+ diff & patch

How it works

+ Commits

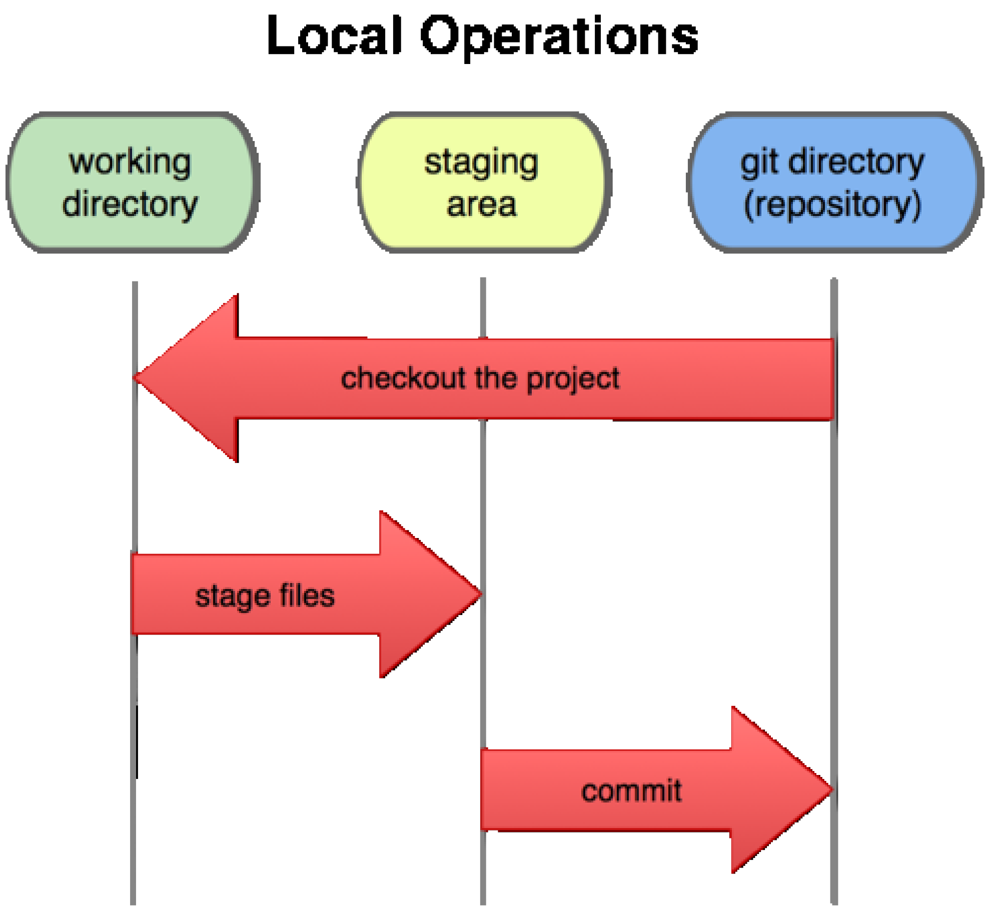
+ sha1

```
(sha1 (open-input-string "hello world"))
"2aae6c35c94fcfb415dbe95f408b9ce91ee846ed"
```

+ diff & patch

+ working directory | index | repository

That working directory | index | repository thing



3: Try it out

Git Commands

`git config`

Get and set repository or global options.

```
git config --global user.name "Jordan Schatz"
git config --global user.email "jordan@noionlabs.com"
```

Git Commands

`git init`

Create an empty git repository.

Git Commands

`git add` Add file contents to the index.

Git Commands

`git status` Show the working tree status.

Git Commands

`git commit` Record changes to the repository.

Git Commands

`gitk` The git repository browser.

Git Commands

`git log` Show commit logs.

Git Commands

`git clone`

Clone a repository into a new directory.

Git Commands

`git branch` List, create, or delete branches.

Git Commands

`git checkout`

Checkout a branch or paths to the working tree.

Git Commands

`git push`

Update remote refs along with associated objects.

Git Commands

`git fetch`

Download objects and refs from another repository.

Git Commands

`git pull`

Fetch from and merge with another repository or a local branch.

Git Commands

`git remote` manage set of tracked repositories.

Git Commands

`git stash`

Stash the changes in a dirty working directory away.

Git Commands

`git diff`

Show changes between commits, commit and working tree, etc.

Git Commands

`git clean`

Remove untracked files from the working tree.

Git Commands

`git merge`

Join two or more development histories together.

Git Commands

`git rebase`

Forward-port local commits to the updated upstream head.

Git Commands

`git fsck`

Verifies the connectivity and validity of the objects in the database.

Git Commands

```
git gc
```

Cleanup unnecessary files and optimize the local repository.

Git Commands

`git prune`

Prune all unreachable objects from the object database.

Git Commands

`git tag`

Create, list, delete or verify a tag object signed with GPG.

Git Commands

`git grep` Print lines matching a pattern.

Git Commands

`git blame`

Show what revision and author last modified each line of a file.

Git Commands

`git gui` A portable graphical interface to Git.

Git Commands

`git cherry-pick`

Apply the changes introduced by some existing commits.

Git Commands

`git bisect`

Find by binary search the change that introduced a bug.

Git Commands

`git show` Show various types of objects.

Git Commands

`git format-patch`

Prepare patches for e-mail submission.

Git Commands

`git am` Apply a series of patches from a mailbox.

Git Commands

`git reset`

Reset current HEAD to the specified state.

Git Commands

`git rm`

Remove files from the working tree and from the index.

Git Commands

`git archive`

Create an archive of files from a named tree.

Git Commands

`git mv`

Move or rename a file, a directory, or a symlink.

Git Commands

`git revert` Revert some existing commits.

Git Commands

`git shortlog` Summarize git log output.

Addendum

Gotcha’s

+ Git tracks content, not files.

Gotcha’s

+ Git tracks content, not files.
+ .gitignore

Gotcha’s

+ Git tracks content, not files.
+ .gitignore
+ git push (it does the right thing but)

Gotcha’s

+ Git tracks content, not files.
+ .gitignore
+ git push (it does the right thing but)
+ making commits/checkouts as root

Gotcha’s

- Git tracks content, not files.
- .gitignore
- git push (it does the right thing but)
- making commits/checkouts as root
- chmod’ing the hooks when you didn’t mean too...

Tools

- Github https://github.com/

Tools

+ Github https://github.com/
+ Deploy HQ http://www.deployhq.com/

Tools

- Github https://github.com/
- Deploy HQ http://www.deployhq.com/
- Gource http://code.google.com/p/gource/

Tools

+ Github https://github.com/
+ Deploy HQ http://www.deployhq.com/
+ Gource http://code.google.com/p/gource/
+ Gitosis

http://scie.nti.st/2007/11/14/
hosting-git-repositories-the-easy-and-secure-way