Git Under the Hood

Commits & trees & blobs! Oh My!

Overview

- History
- Basics
- Git Object Model
- References

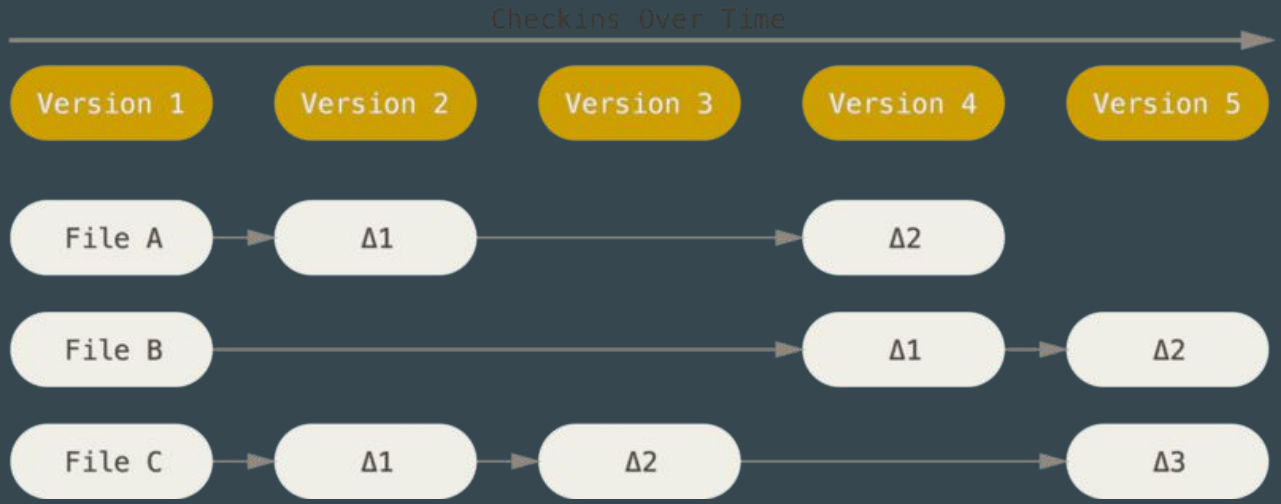
Short History

- < 2005: BitKeeper manages Linux kernel
  - Proprietary DVCS with special Freeware licence
  - Community unhappy
  - Monotone project has nice features but performance is terrible
- April 2005: Git development begins with goals:
  - Speed
  - Simple design
  - Strong support for non-linear development (thousands of parallel branches)
  - Fully distributed
  - Able to handle large projects like the Linux kernel efficiently (speed and data size)
- Self-hosting after ~1 month
- Git v1.0 released December 2005

Source: https://git.wiki.kernel.org/index.php/GitHistory

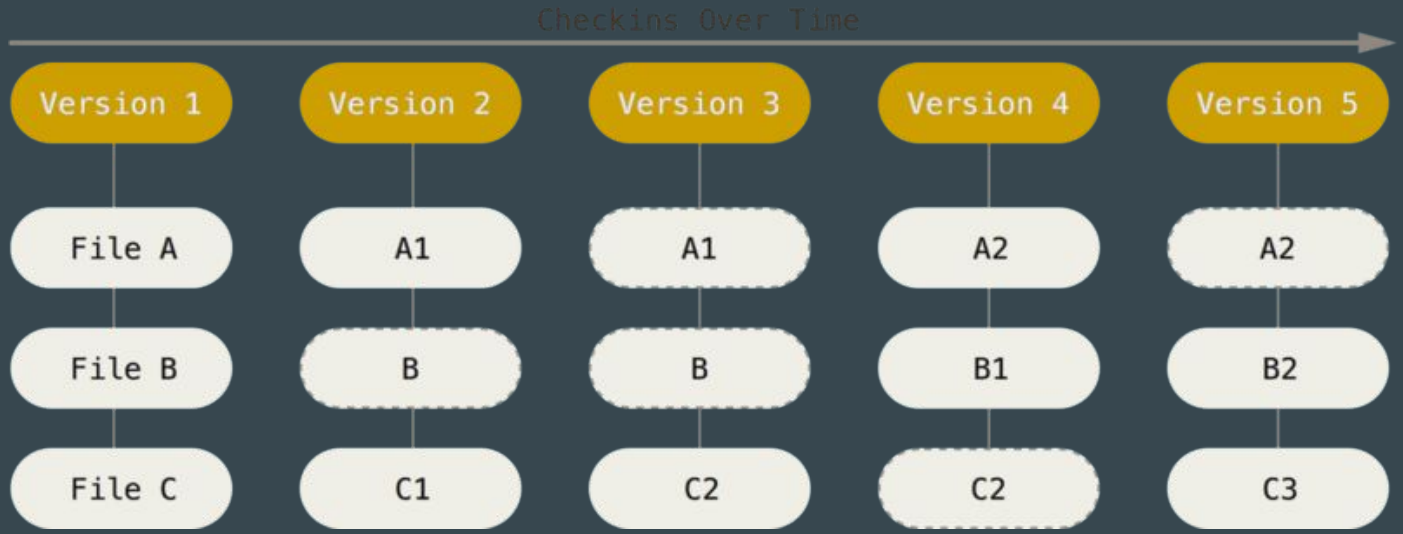
Basics

Traditional VCS - File-based deltas



Basics

Git - Stream of Snapshots

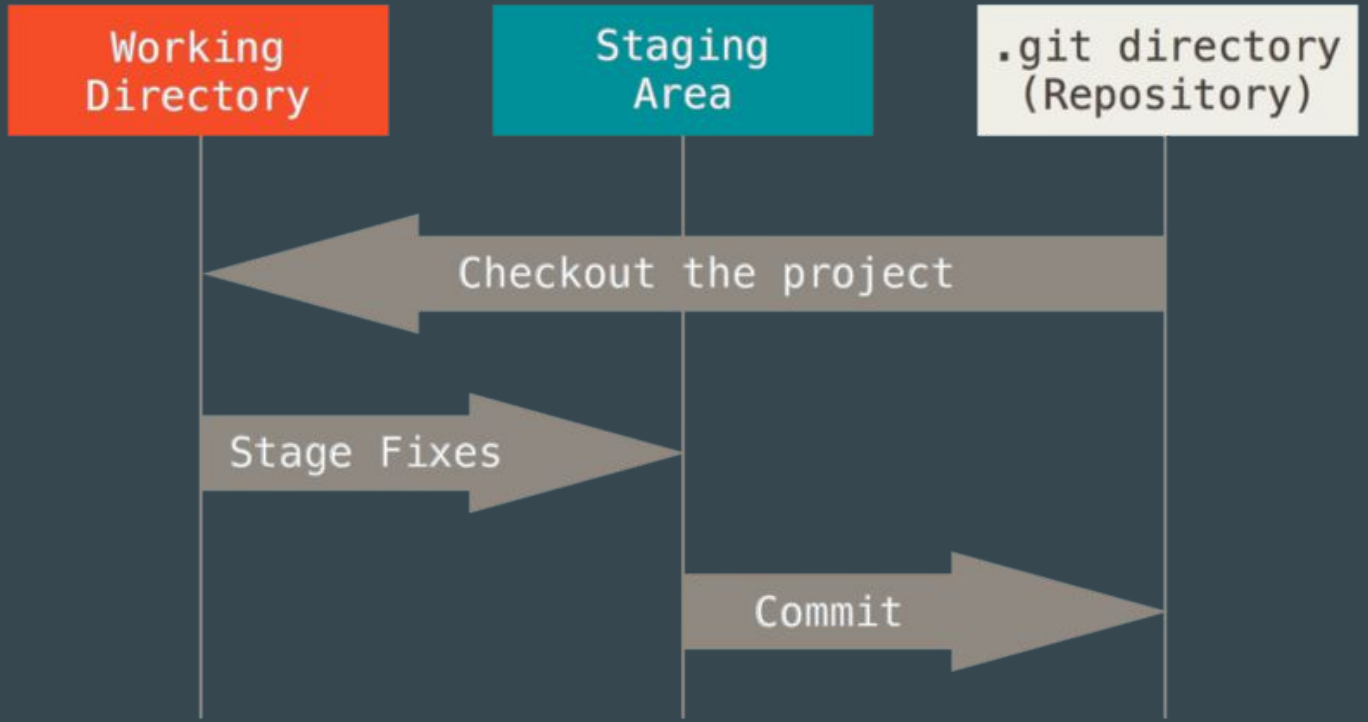


- Nearly every operation is local → Speed

Integrity

- Every object in Git is checksummed
  - impossible to change the contents of any file or directory without Git knowing about it
- SHA-1 hash, 20-byte (160 bits) digest
- Collision probability:
  - *Here's an example to give you an idea of what it would take to get a SHA-1 collision. If all 6.5 billion humans on Earth were programming, and every second, each one was producing code that was the equivalent of the entire Linux kernel history (3.6 million Git objects) and pushing it into one enormous Git repository, it would take roughly 2 years until that repository contained enough objects to have a 50% probability of a single SHA-1 object collision. A higher probability exists that every member of your programming team will be attacked and killed by wolves in unrelated incidents on the same night.*

3 States of Git



- Checkout from `.git` repository
- Changes to current snapshot added to stage
- Commit changes in stage to repository

.git Directory

- `git init` creates a new repository

```
$ ls --classify --format=single-column .git
HEAD
config
description
hooks/
info/
objects/
refs/
```

- config - project specific settings
- info - directory for project-specific excludes
- Leaves `HEAD`, `objects` & `refs` directories

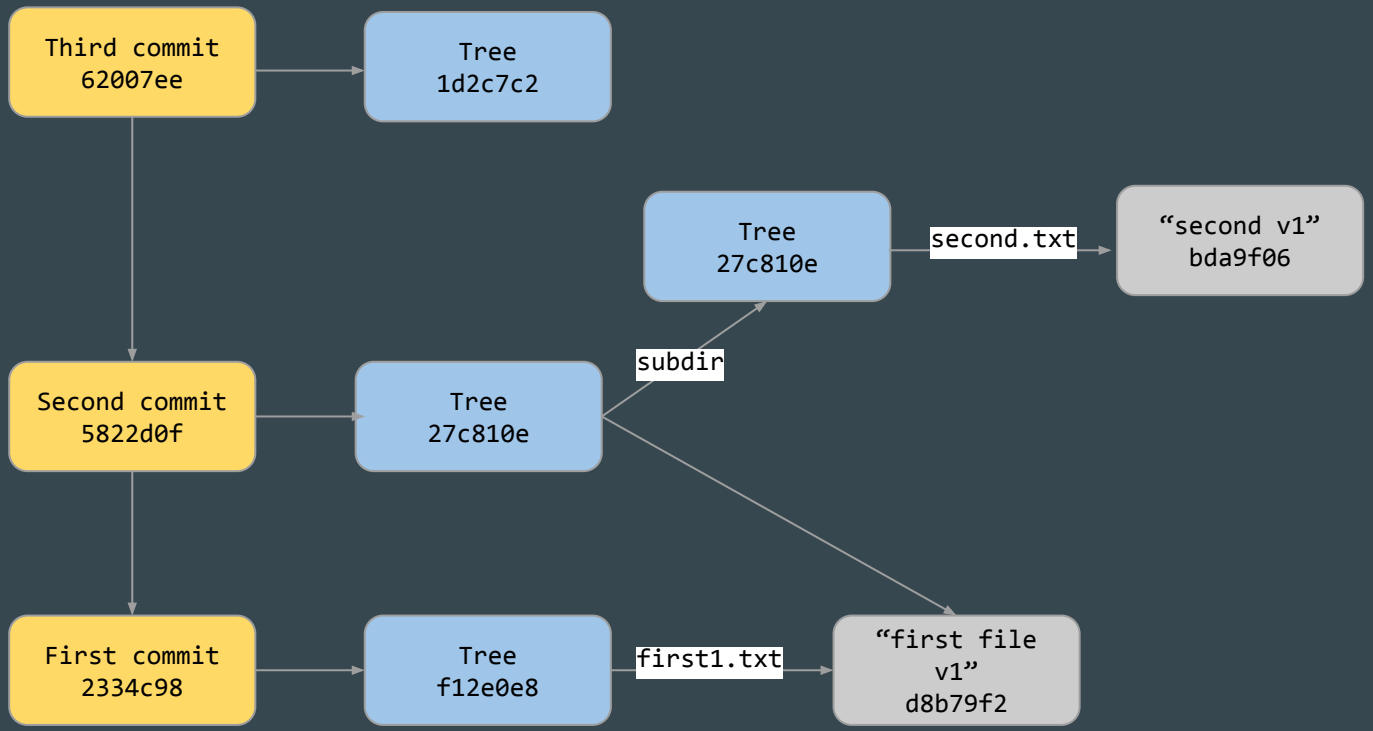
Git Object Model

- Core of Git is a simple key-value store
  - The key is the SHA-1 hash and the value is the hashed content

- Content is stored in objects of different types representing the different concepts in the system:
  - *blob (binary large object)*
  - *tree*
  - *commit*
  - *annotated tag (not going to cover this)*
- Every object is referred to by its hash and is found in the `.git/objects` directory

→ *DEMO!*

Graph


Third commit
62007ee
Tree
1d2c7c2
Tree
27c810e
second.txt
“second v1”
bda9f06
Second commit
5822d0f
Tree
27c810e
subdir
First commit
2334c98
Tree
f12e0e8
first1.txt
“first file
v1”
d8b79f2

References

- All Git operations can be performed using the SHA1 values
  - This is very inconvenient!
  - We would rather refer to things by an alias
- Enter “references” and the `.git/refs` directory and the special reference `.git/HEAD`

- BACK TO DEMO!

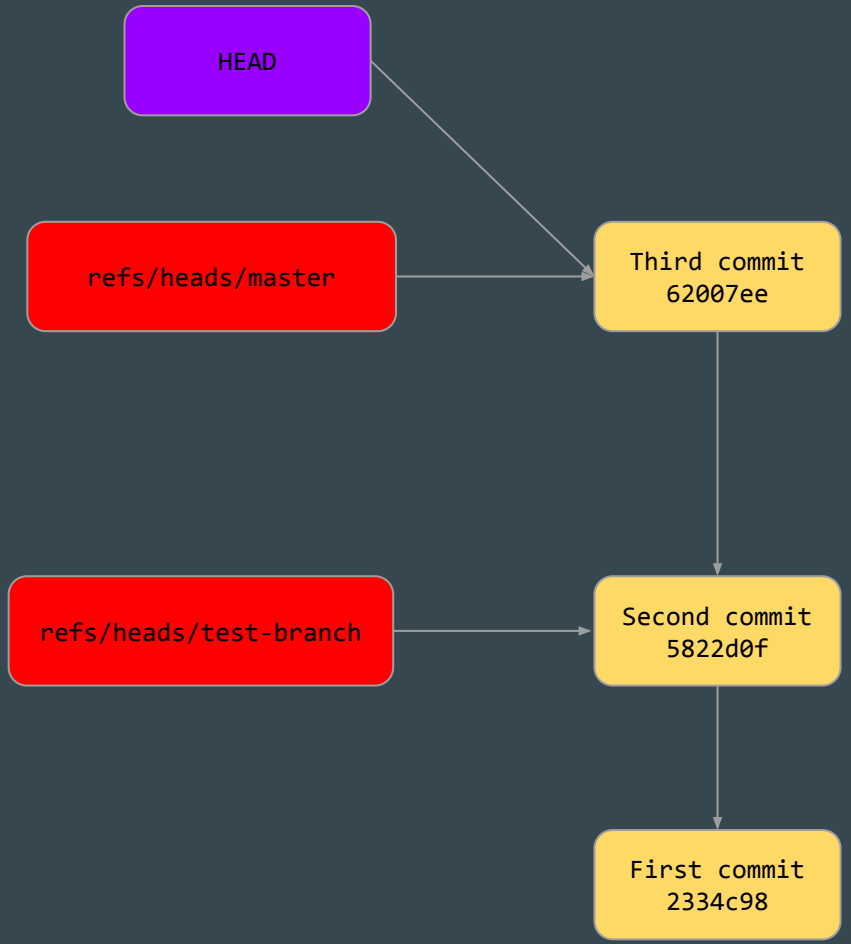
Branches


HEAD
refs/heads/master
Third commit
62007ee
refs/heads/test-branch
Second commit
5822d0f
First commit
2334c98

Reflog

- Useful command: `git reflog`
- Shows the history of where the HEAD pointer has been

Detached HEAD



HEAD

Who has the first question?