

# Understanding Git

---

Sylvain Bouveret, Grégory Mounié, Matthieu Moy  
2021

[first].[last]@imag.fr

[https://git.pages.ensimag.fr/formation-git/slides/  
understanding-git-slides.pdf](https://git.pages.ensimag.fr/formation-git/slides/understanding-git-slides.pdf)



- Presenting the data model behind Git
- Showing how Git stores data and history
- Understanding how to navigate between the commits of a repository

THIS IS GIT. IT TRACKS COLLABORATIVE WORK  
ON PROJECTS THROUGH A BEAUTIFUL  
DISTRIBUTED GRAPH THEORY TREE MODEL.

COOL. HOW DO WE USE IT?

NO IDEA. JUST MEMORIZE THESE SHELL  
COMMANDS AND TYPE THEM TO SYNC UP.  
IF YOU GET ERRORS, SAVE YOUR WORK  
ELSEWHERE, DELETE THE PROJECT,  
AND DOWNLOAD A FRESH COPY.







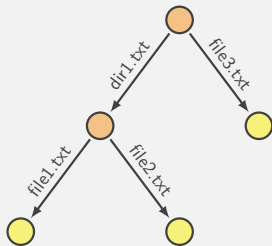
If that doesn't fix it, `git.txt` contains the phone number of a friend of mine who understands git. Just wait through a few minutes of "It's really pretty simple, just think of branches as..." and eventually you'll learn the commands that will fix everything.




- Beauty of Git: **very** simple data model  
(The tool is clever, the repository format is simple&stupid)
- Understand the model, and the 150+ commands will become **simple!**

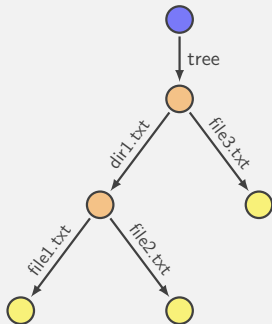
## Objects, sha1

---




-  blob Any sequence of bytes, represents file content
-  tree Associates object to pathnames, represents a directory

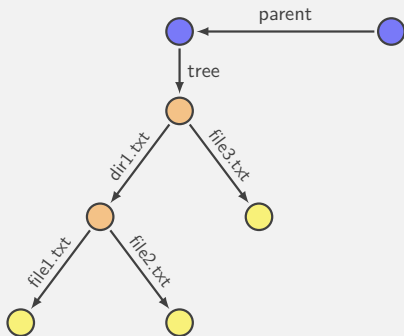





-  blob Any sequence of bytes, represents file content
-  tree Associates object to pathnames, represents a directory
-  commit Metadata + pointer to tree + pointer to parents

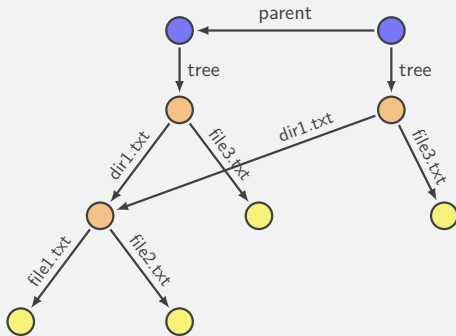







-  blob Any sequence of bytes, represents file content
-  tree Associates object to pathnames, represents a directory
-  commit Metadata + pointer to tree + pointer to parents

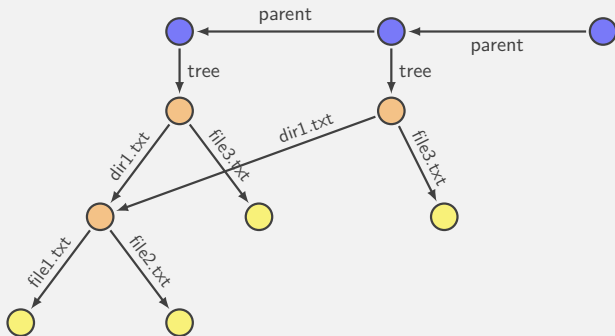


-  blob Any sequence of bytes, represents file content
-  tree Associates object to pathnames, represents a directory
-  commit Metadata + pointer to tree + pointer to parents






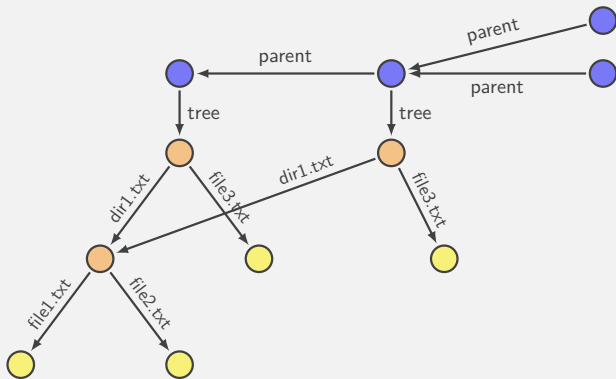
# Content of a Git repository: Git objects

-  blob Any sequence of bytes, represents file content
-  tree Associates object to pathnames, represents a directory
-  commit Metadata + pointer to tree + pointer to parents






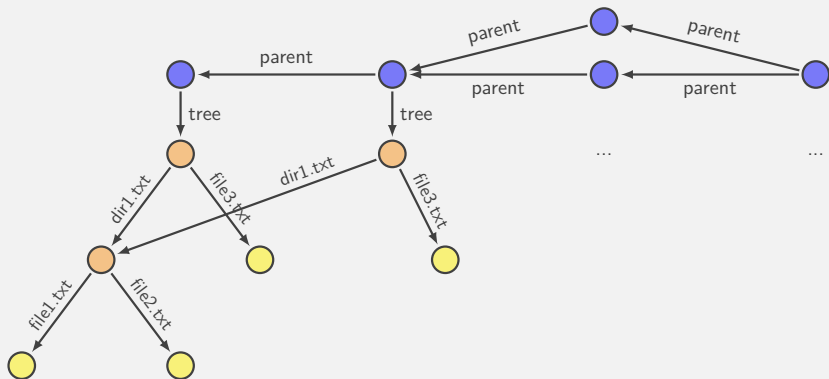
# Content of a Git repository: Git objects

-  blob Any sequence of bytes, represents file content
-  tree Associates object to pathnames, represents a directory
-  commit Metadata + pointer to tree + pointer to parents



# Content of a Git repository: Git objects

-  blob Any sequence of bytes, represents file content
-  tree Associates object to pathnames, represents a directory
-  commit Metadata + pointer to tree + pointer to parents



```
$ git log
commit 7a7fb77be431c284f1b6d036ab9aebf646060271
Author: Matthieu Moy <Matthieu.Moy@imag.fr>
Date:   Wed Jul 2 20:13:49 2014 +0200

    Initial commit
$ find .git/objects/
.git/objects/
.git/objects/fc
.git/objects/fc/264b697de62952c9ff763b54b5b11930c9cfec
.git/objects/7a
.git/objects/7a/7fb77be431c284f1b6d036ab9aebf646060271
.git/objects/50
.git/objects/50/a345788a8df75e0f869103a8b49cecdf95a416
.git/objects/26
.git/objects/26/27a0555f9b58632be848fee8a4602a1d61a05f
```

# Git objects: On-disk format

```
$ echo foo > README.txt; git add README.txt
$ git commit -m "add README.txt"
```

```
[master 5454e3b] add README.txt
 1 file changed, 1 insertion(+)
 create mode 100644 README.txt
$ find .git/objects/
.git/objects/
.git/objects/fc
.git/objects/fc/264b697de62952c9ff763b54b5b11930c9cfec
.git/objects/7a
.git/objects/7a/7fb77be431c284f1b6d036ab9aebf646060271
.git/objects/25
.git/objects/25/7cc5642cb1a054f08cc83f2d943e56fd3ebe99
.git/objects/54
.git/objects/54/54e3b51e81d8d9b7e807f1fc21e618880c1ac9
...
```

- By default, 1 object = 1 file
- Name of the file = object unique identifier content
- Content-addressed database:
  - Identifier computed as a hash of its content
  - Content accessible from the identifier
- Consequences:
  - Objects are immutable
  - Objects with the same content have the same identity (deduplication for free)
  - Previously, no known collision in SHA1, now moving to SHA-256
  - Acyclic (DAG = Directed Acyclic Graph)



```
$ du -sh .git/objects/  
68K      .git/objects/  
$ git gc  
...  
$ du -sh .git/objects/  
24K      .git/objects/  
$ find .git/objects/  
.git/objects/  
.git/objects/pack  
.git/objects/pack/pack-f9cbdc53005a4b500934625d...a3.idx  
.git/objects/pack/pack-f9cbdc53005a4b500934625d...a3.pack  
.git/objects/info  
.git/objects/info/packs  
$
```

↪ More efficient format, no conceptual change  
(objects are still there)

- `git cat-file -p` : pretty-print the content of an object

```
$ git log --oneline
5454e3b add README.txt
7a7fb77 Initial commit

$ git cat-file -p 5454e3b
tree 59802e9b115bc606b88df4e2a83958423661d8c4
parent 7a7fb77be431c284f1b6d036ab9aebf646060271
author Matthieu Moy <Matthieu.Moy@imag.fr> 1404388746 +0200
committer Matthieu Moy <Matthieu.Moy@imag.fr> 1404388746 +0200

add README.txt
```

- `git cat-file -p` : pretty-print the content of an object

```
$ git cat-file -p 59802e9b115bc606b88df4e2a83958423661d8c4
100644 blob 257cc5642cb1a054f08cc83f2d943e56fd3ebe99 README.txt
040000 tree 2627a0555f9b58632be848fee8a4602a1d61a05f sandbox
```

```
$ git cat-file -p 257cc5642cb1a054f08cc83f2d943e56fd3ebe99
foo
```

```
$ printf 'blob 4\0foo\n' | shasum
257cc5642cb1a054f08cc83f2d943e56fd3ebe99 -
```

```
$ git checkout -b branch HEAD^
Switched to a new branch 'branch'
$ echo foo > file.txt; git add file.txt
$ git commit -m "add file.txt"
[branch f44e9ab] add file.txt
1 file changed, 1 insertion(+)
create mode 100644 file.txt
$ git merge master
Merge made by the 'recursive' strategy.
README.txt | 1 +
1 file changed, 1 insertion(+)
create mode 100644 README.txt
```

```
$ git checkout -b branch HEAD~
$ echo foo > file.txt; git add file.txt
$ git commit -m "add file.txt"
$ git merge master
$ git log --oneline --graph
* 1a7f9ae (HEAD, branch) Merge branch 'master' into branch>
|\
| * 5454e3b (master) add README.txt
* | f44e9ab add file.txt
|/
* 7a7fb77 Initial commit
$ git cat-file -p 1a7f9ae
tree 896dbd61ffc617b89eb2380cdcaffcd7c7b3e183
parent f44e9abff8918f08e91c2a8fefe328dd9006e242
parent 5454e3b51e81d8d9b7e807f1fc21e618880c1ac9
author Matthieu Moy Matthieu.Moy@imag.fr 1404390461 +0200
committer Matthieu Moy Matthieu.Moy@imag.fr 1404390461 +0200

Merge branch 'master' into branch
```

- A commit represents **exactly** the state of the project
- A tree represents **only** the state of the project (where we are, not how we got there)
- Renames are not tracked, but re-detected on demand
- Diffs are computed on demand (e.g. `git diff HEAD HEAD^`)
- Physical storage still efficient

# References

---

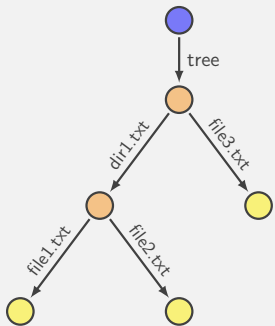
- In Java:

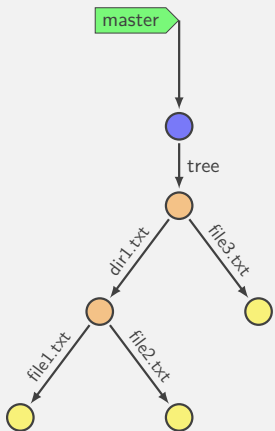
```
String s; // Reference named s
s = new String("foo"); // Object pointed to by s
String s2 = s; // Two refs for the same object
```

- In Git: likewise!

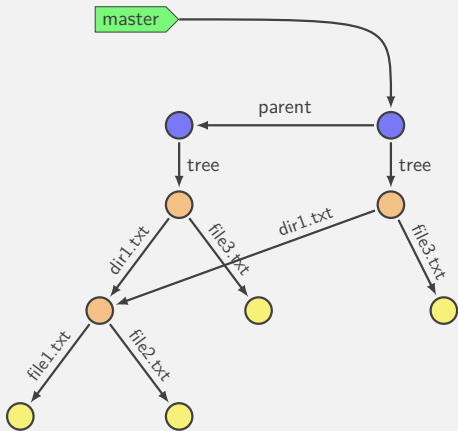
```
$ git log --oneline
5454e3b add README.txt
7a7fb77 Initial commit
$ cat .git/HEAD
ref: refs/heads/master
$ cat .git/refs/heads/master
5454e3b51e81d8d9b7e807f1fc21e618880c1ac9
$ git symbolic-ref HEAD
refs/heads/master
$ git rev-parse refs/heads/master
5454e3b51e81d8d9b7e807f1fc21e618880c1ac9
```



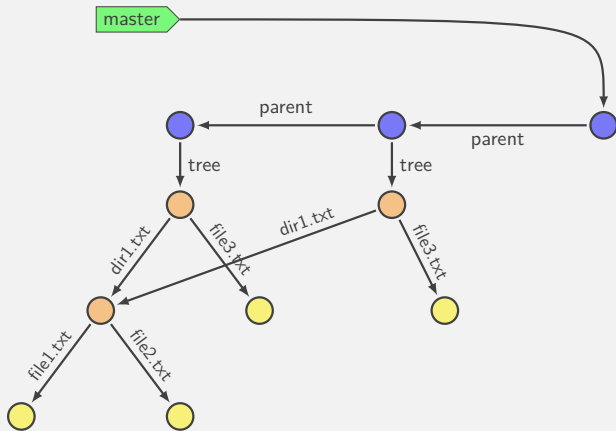




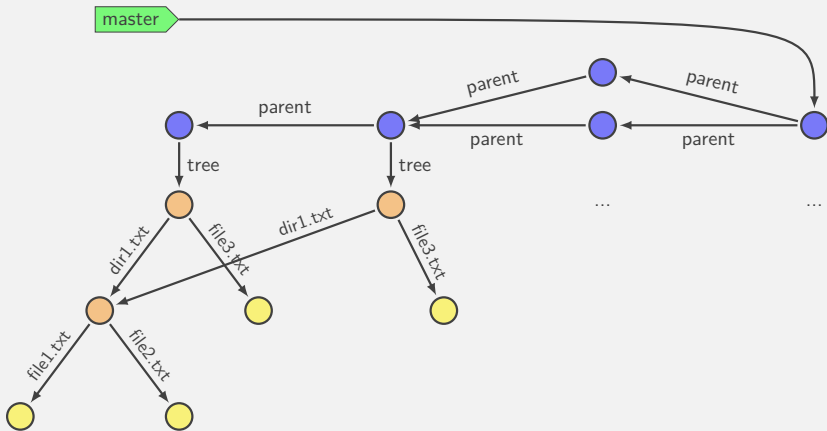
# References (refs) and objects



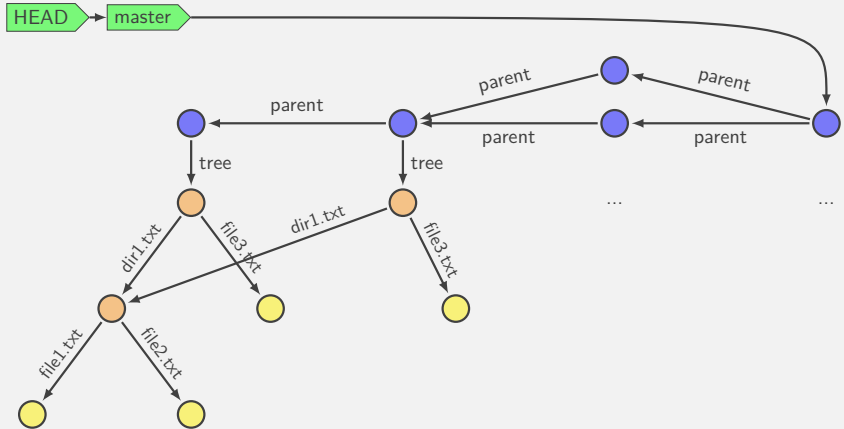
# References (refs) and objects



# References (refs) and objects



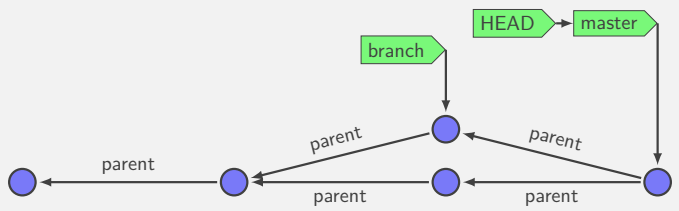
# References (refs) and objects



File Edit View Help

master	Merge branch 'branch'	Matthieu Moy <Matthieu.Moy@	2014-07-03 18:05:56
branch	CCC	Matthieu Moy <Matthieu.Moy@	2014-07-03 18:05:45
BBB		Matthieu Moy <Matthieu.Moy@	2014-07-03 18:05:35
AAA		Matthieu Moy <Matthieu.Moy@	2014-07-03 18:05:16
Initial commit		Matthieu Moy <Matthieu.Moy@	2014-07-03 18:04:59

SHA1 ID: 23f030117436d69f39690725f140087e26ac59b9



- A branch is a ref to a commit
- A lightweight tag is a ref (usually to a commit) (like a branch, but doesn't move)
- Annotated tags are objects containing a ref + a (signed) message
- HEAD is “where we currently are”
  - If HEAD points to a branch, the next commit will move the branch
  - If HEAD points directly to a commit (detached HEAD), the next commit creates a commit not in any branch (warning!)



## Branches and tags in practice

---

- Create a local branch and check it out:

```
git checkout -b <branch-name>
```

- List local branches:

```
git branch
```

- List all branches (including remote-tracking):

```
git branch -a
```

- Create a tag:

```
git tag <tag-name>
```

- Switch to a branch, a tag, or a commit:

```
git checkout [branch-name | tag-name | commit]
```