

Gitlet Design

Disclaimer

- Will not talk too much about implementation details
- Goal is to be interactive
- Want to hear from you!!!

Agenda

- What to consider when designing?
- What should be a file?
- Common paradigms
 - Read-modify-write
 - Indirection
- Q + A

What to consider?

Gitlet is a Data Structure

All about the storage
of information/data

1. What data?

2. When should we store?

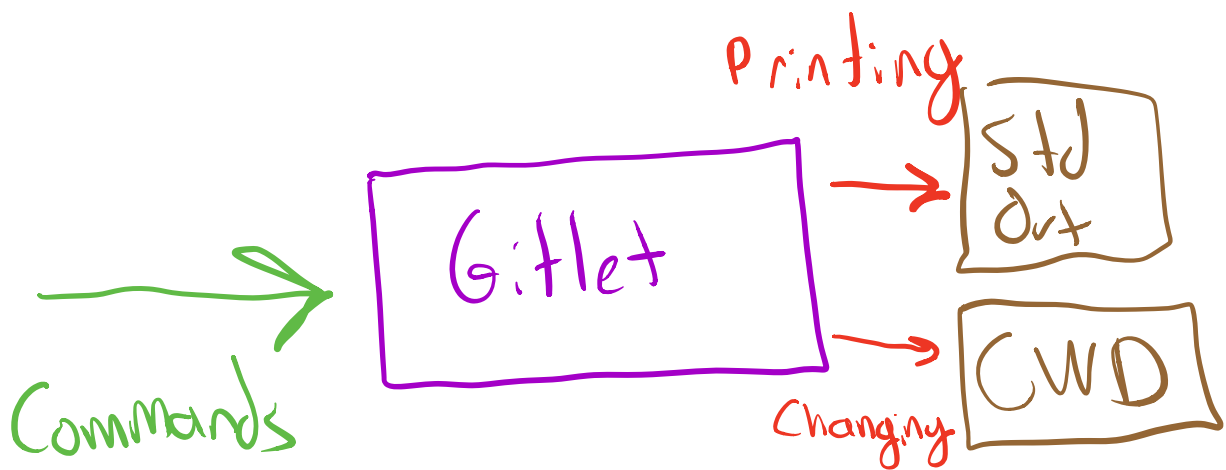
3. How should we store?

We'll do 1, talk about 3, and
I'll leave 2 for you all.

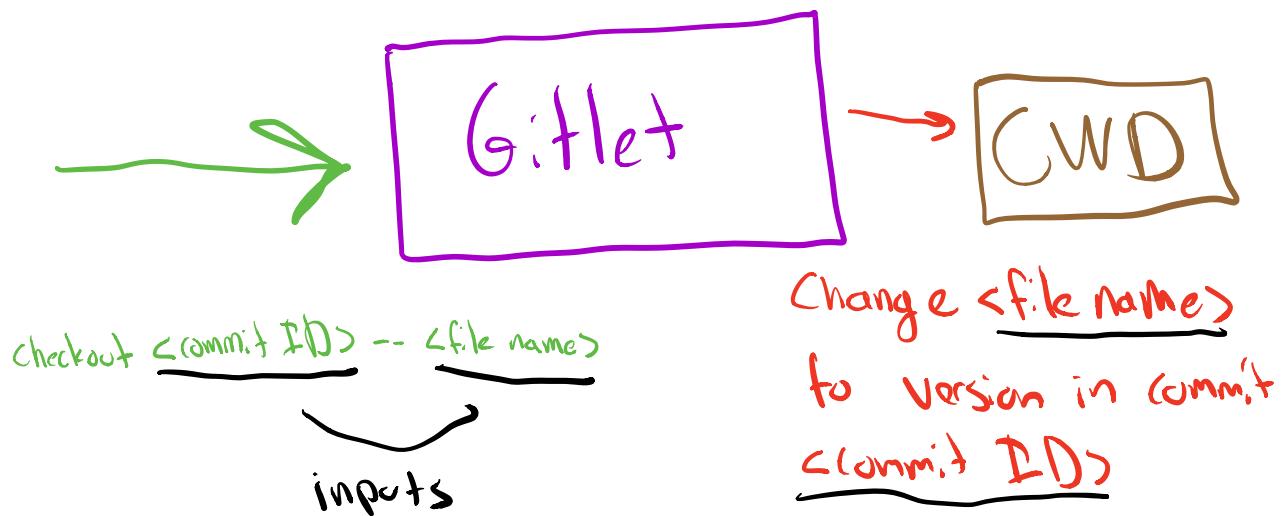
Take a step back and think about Gitlet abstractly in terms of input/output

INPUT

OUTPUT



For example, let's look at the
checkout command



Has multiple implications

1. Need to go from <commit ID> → Commit Object
2. Commits need to convert <file names> → the contents of that file in that commit

Now we know what information needs to be stored to support the checkout command

Still need to figure out:

2. When should this be stored?

3. How should this be stored?

- File?

- Instance var? (What data structure?)

What should be a file

How you store data is a design choice

The 3 ways of storing data:

1. As a static final var
2. As an instance var in a serialized object
3. As a standalone file

If there is only 1 of them,
and it never ever changes, make it
static final

Examples from Capers Lab:

- DOG_FOLDER

- CWD

- STORY

* Static variables aren't serialized

If it's small, ok to make an instance variable in a Serialized class

Examples in Dog from Capers:

- name
- breed
- age

If we made story an instance var

Examples in Commit:

- message
- parent id String parentID
- timestamp

If it could be huge, be safe and
make it a file

Example from Capers
- Story

Can you think of an
example from Git? 3.

- Blob (contents of a file)

Common Paradigms

Read-Modify-Write

Remember the have Birthday command

1. Read in the Dog object
2. Modify the Dog object
3. Write the Dog object back to its file

→
What would happen if we skipped this?

Consider helper functions since you'll do this a lot

Indirection

Why did we make the
Story a file instead of some
instance variable?

Answer:

Space / time efficiency.

That was indirection

It's very simple: just add another pointer

Bad idea

STORY → Contents

Good Idea (with indirection)

STORY → File → Contents

Now your turn

Think about where else
you can apply this logic (in
your head).

Q + A