
Universe2020

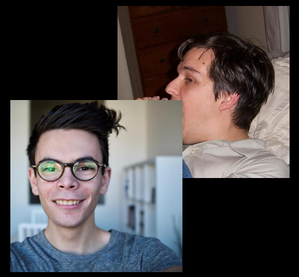
Optimize your monorepo experience

These people make the magic happen

@derrickstolee

Git Client

@dscho
@jeffhostetler
@mjcheetham
@prplr

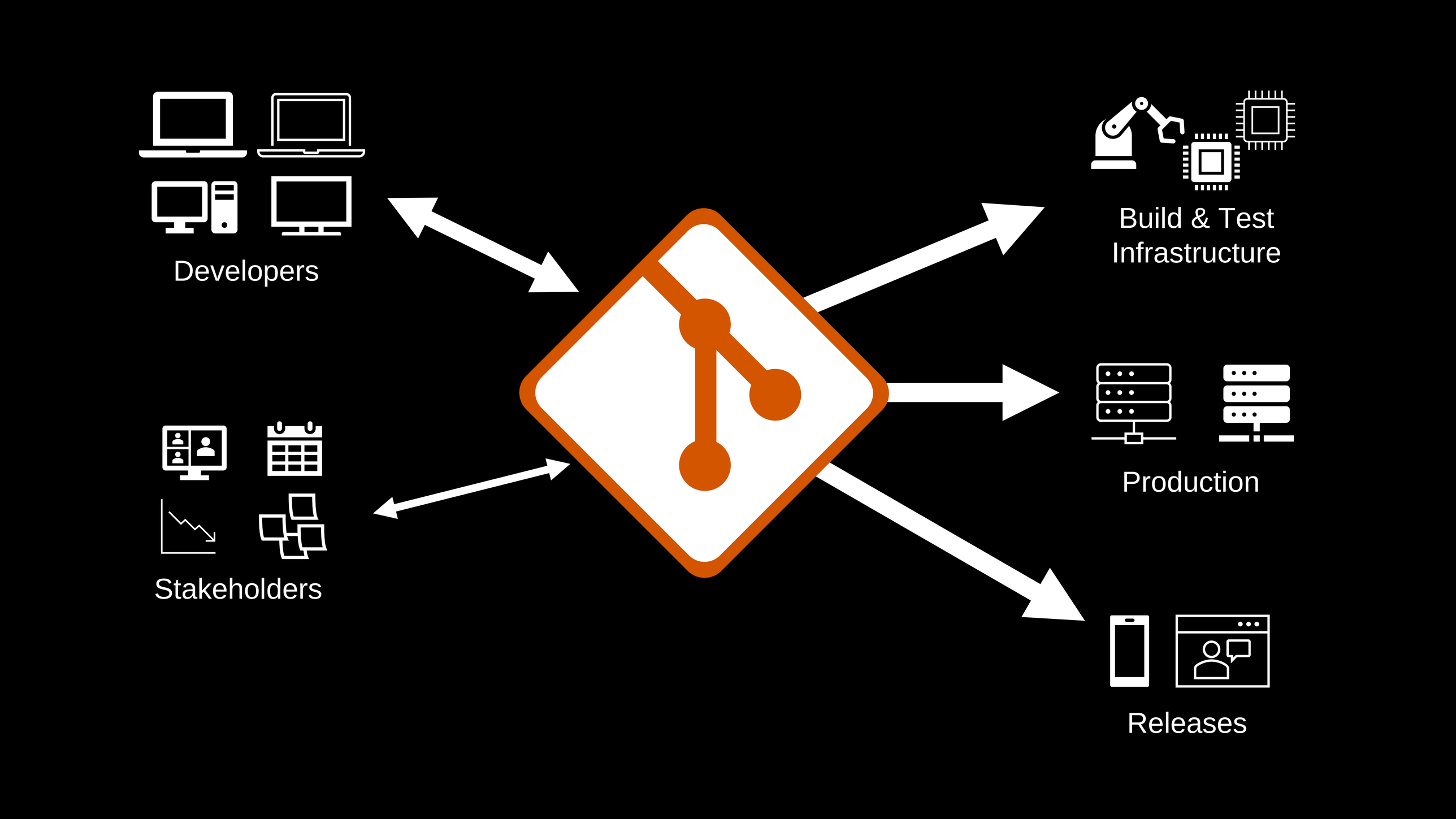


Git Contrib

@peff
@ttaylorr

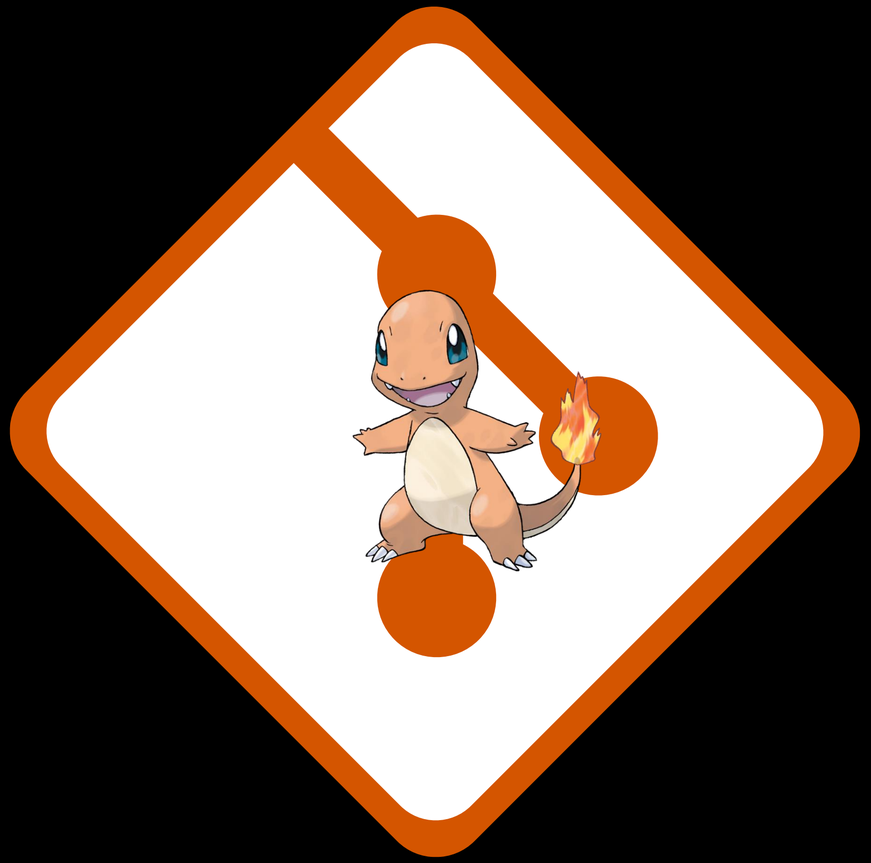
Git Systems

@bk2204
@dkunkler
@mhagger
@wrighty


Developers
Stakeholders
Build & Test
Infrastructure
Production
Releases

How do monorepos grow?

Growing in Git

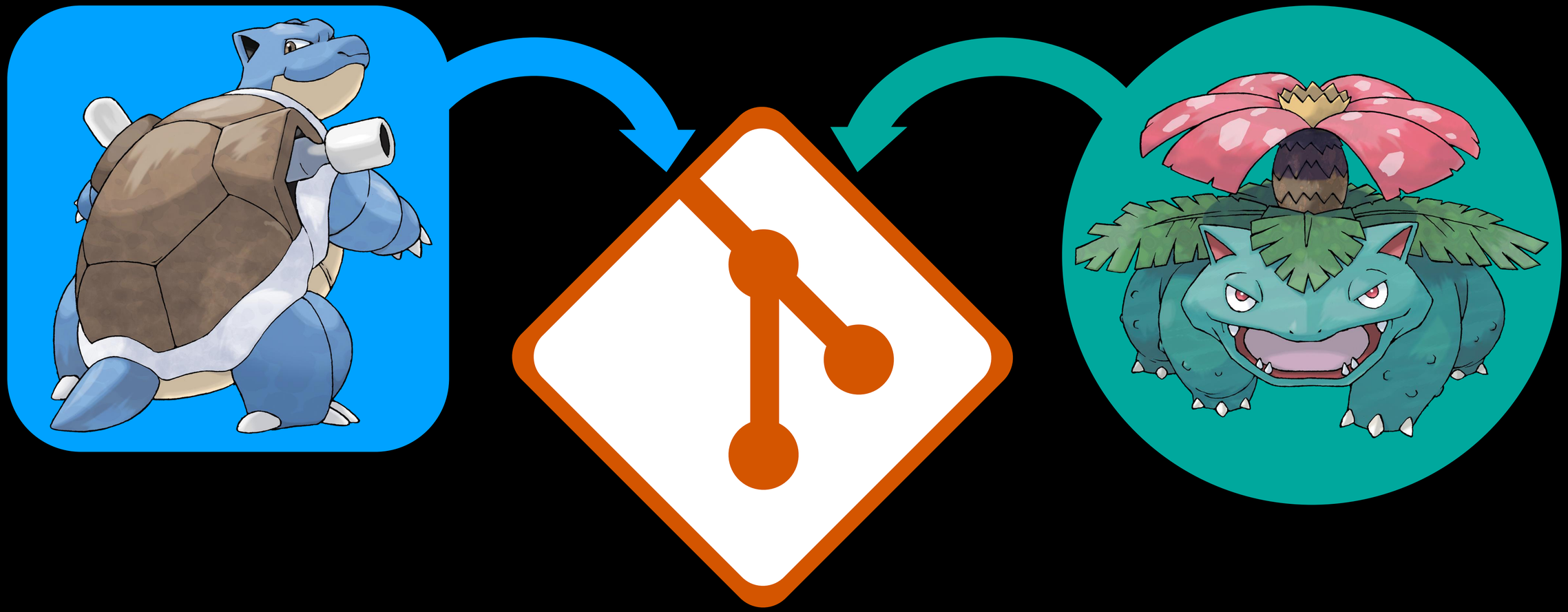


Growing in Git

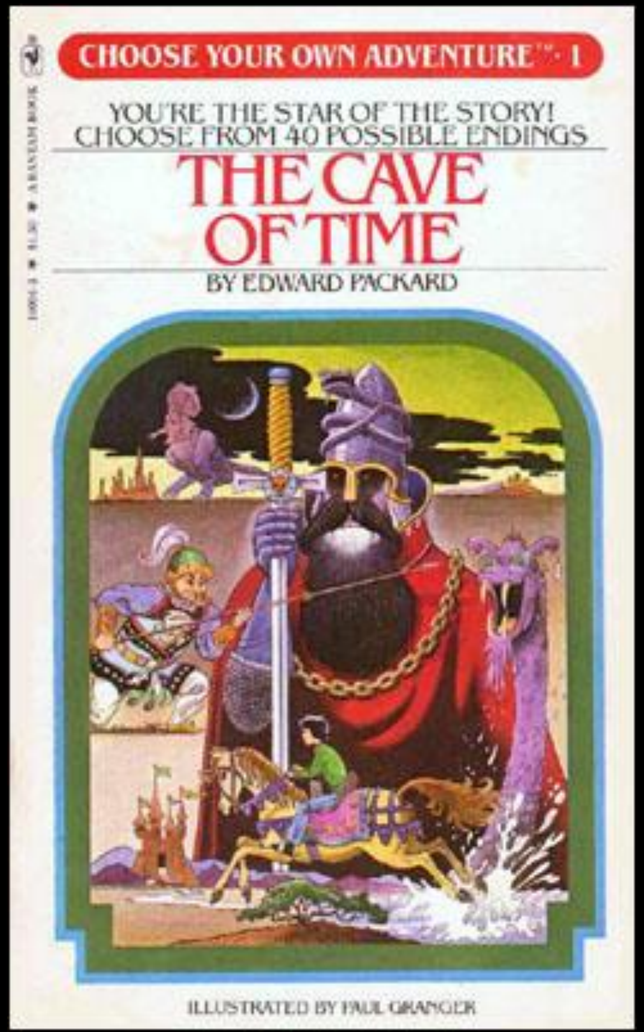
Growing in Git

Growing in Git

Migrating to Git



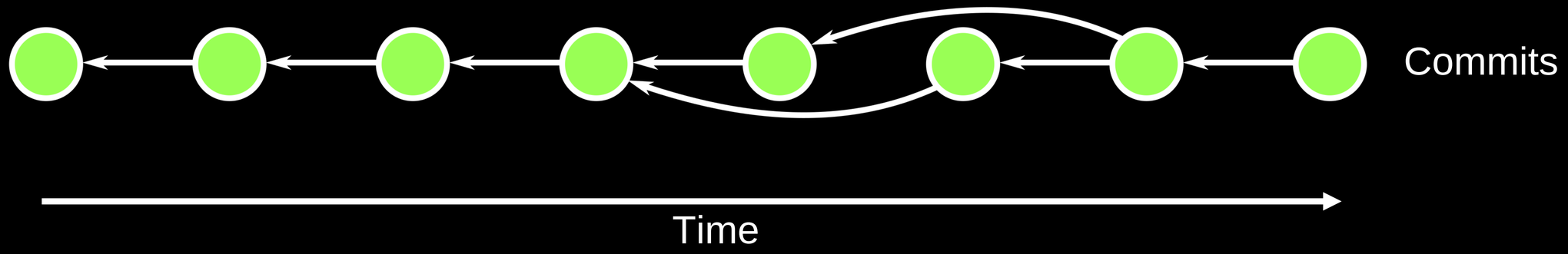
Your choices matter!

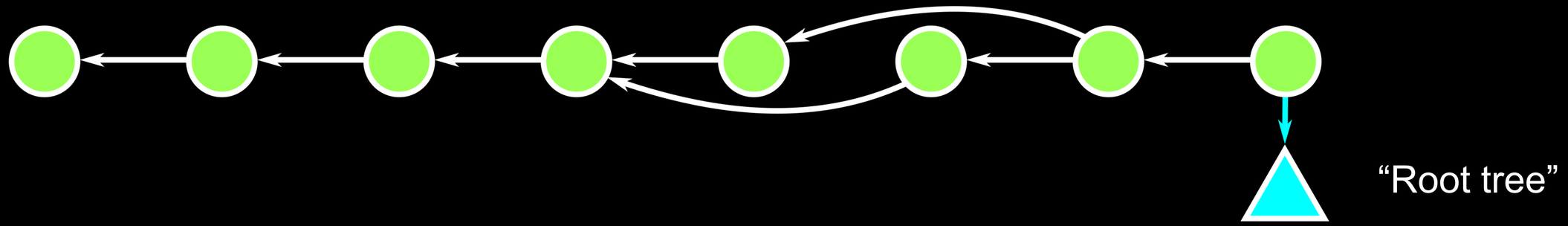


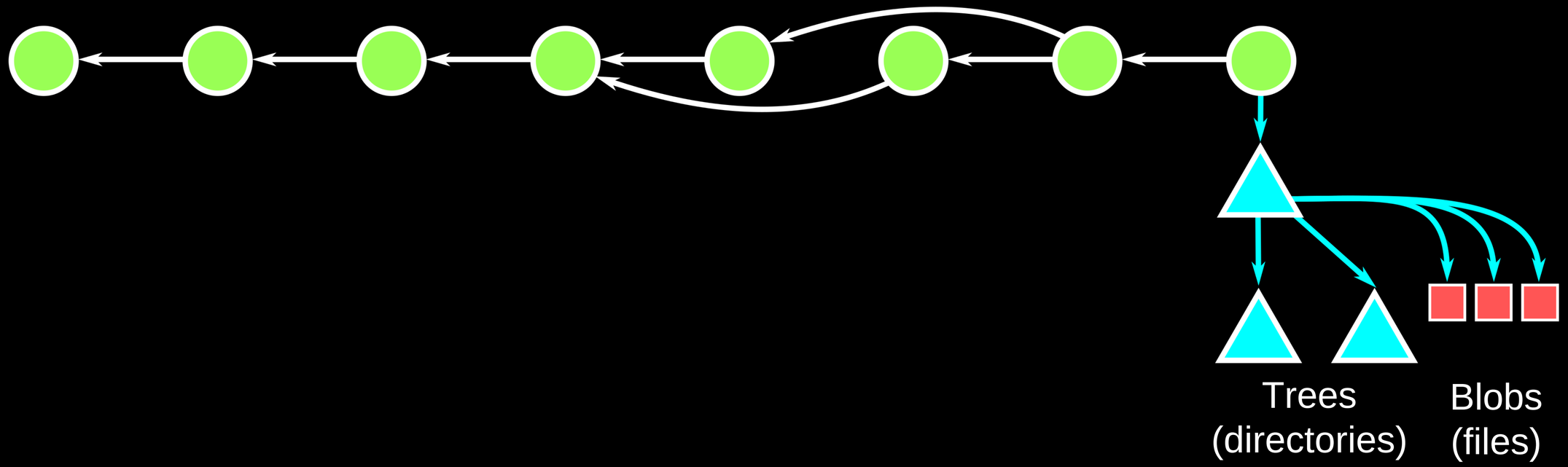
HOT TAKE

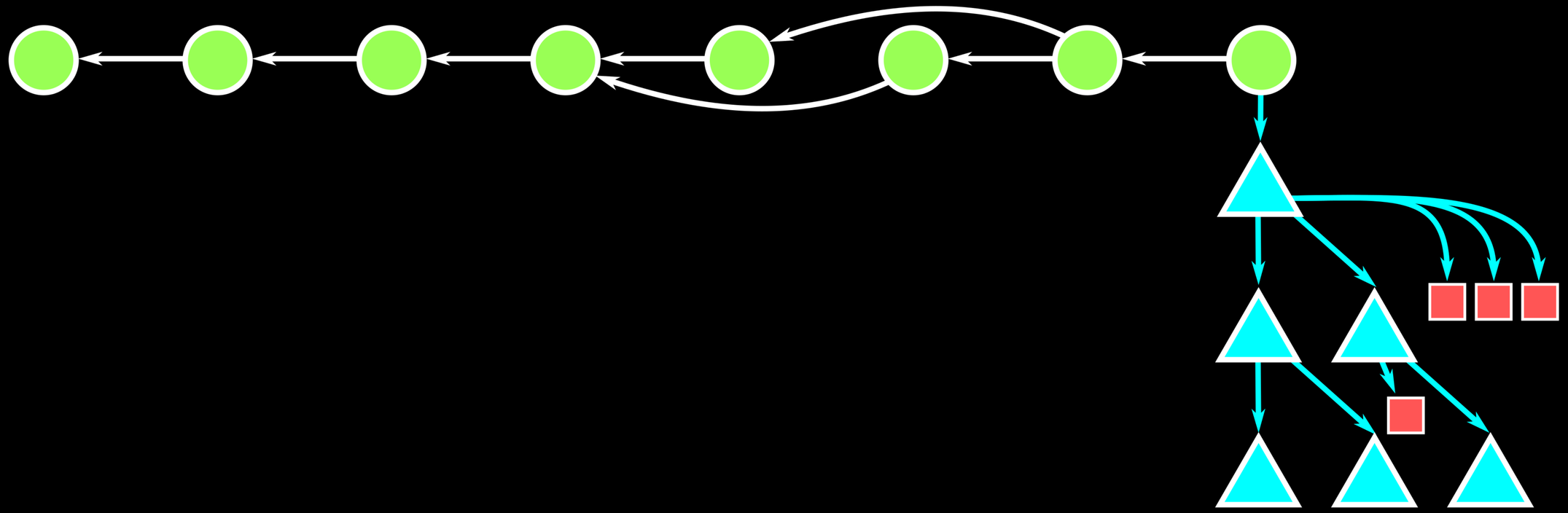
Code never gets *faster*,
it can only do *fewer things*.

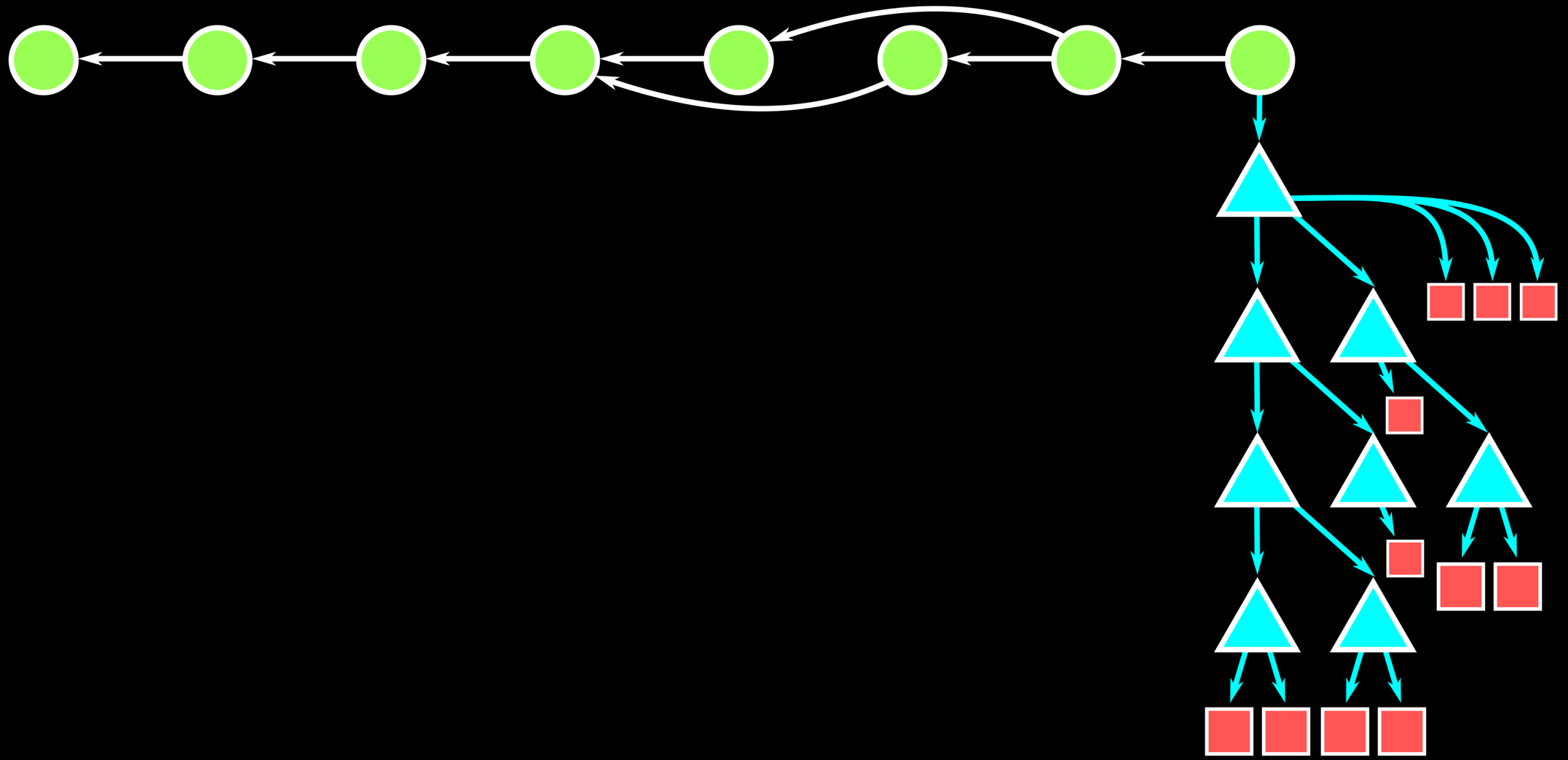

IT'S DANGEROUS TO GO
ALONE! TAKE THIS.

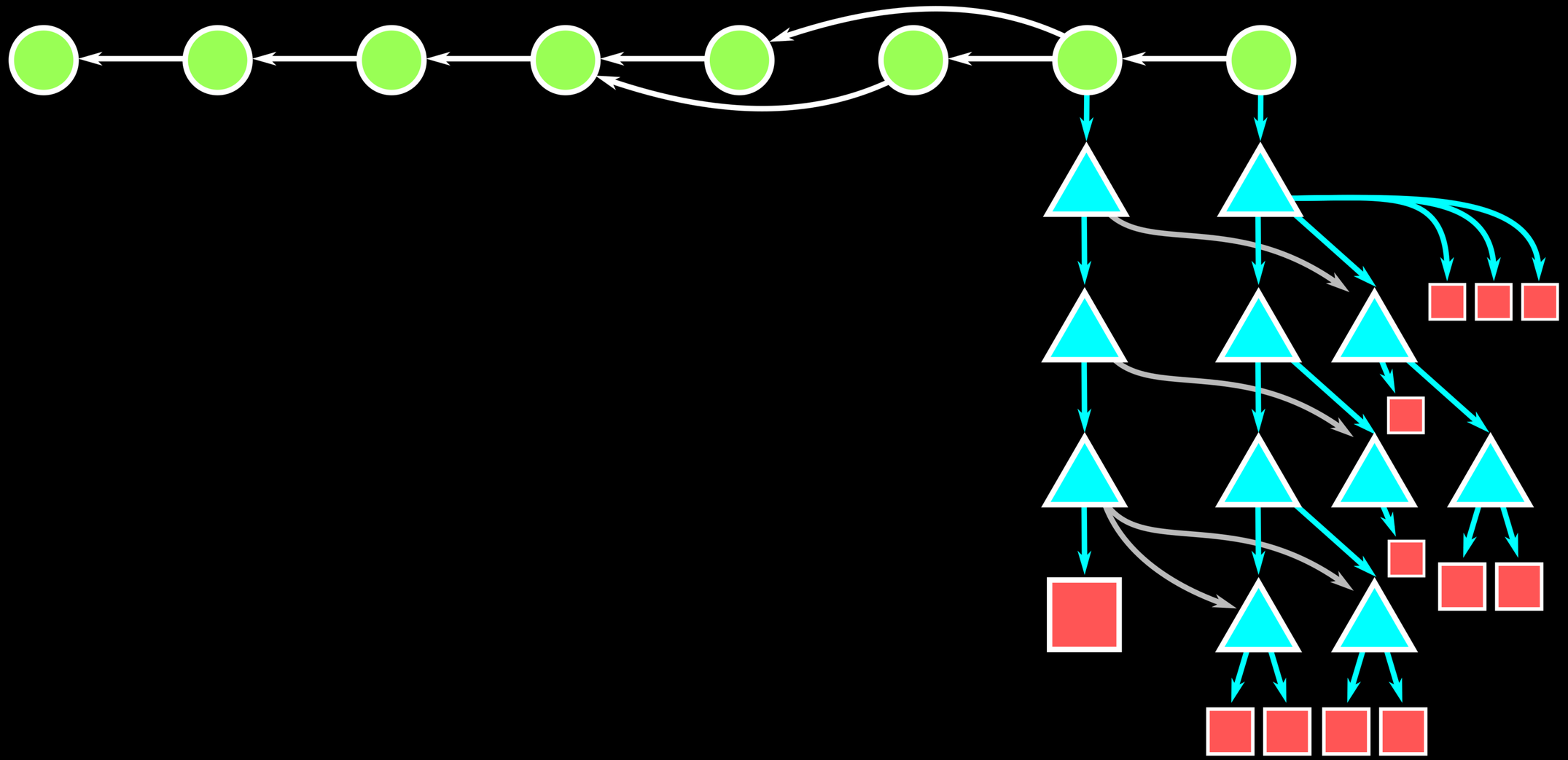

Commits
Time

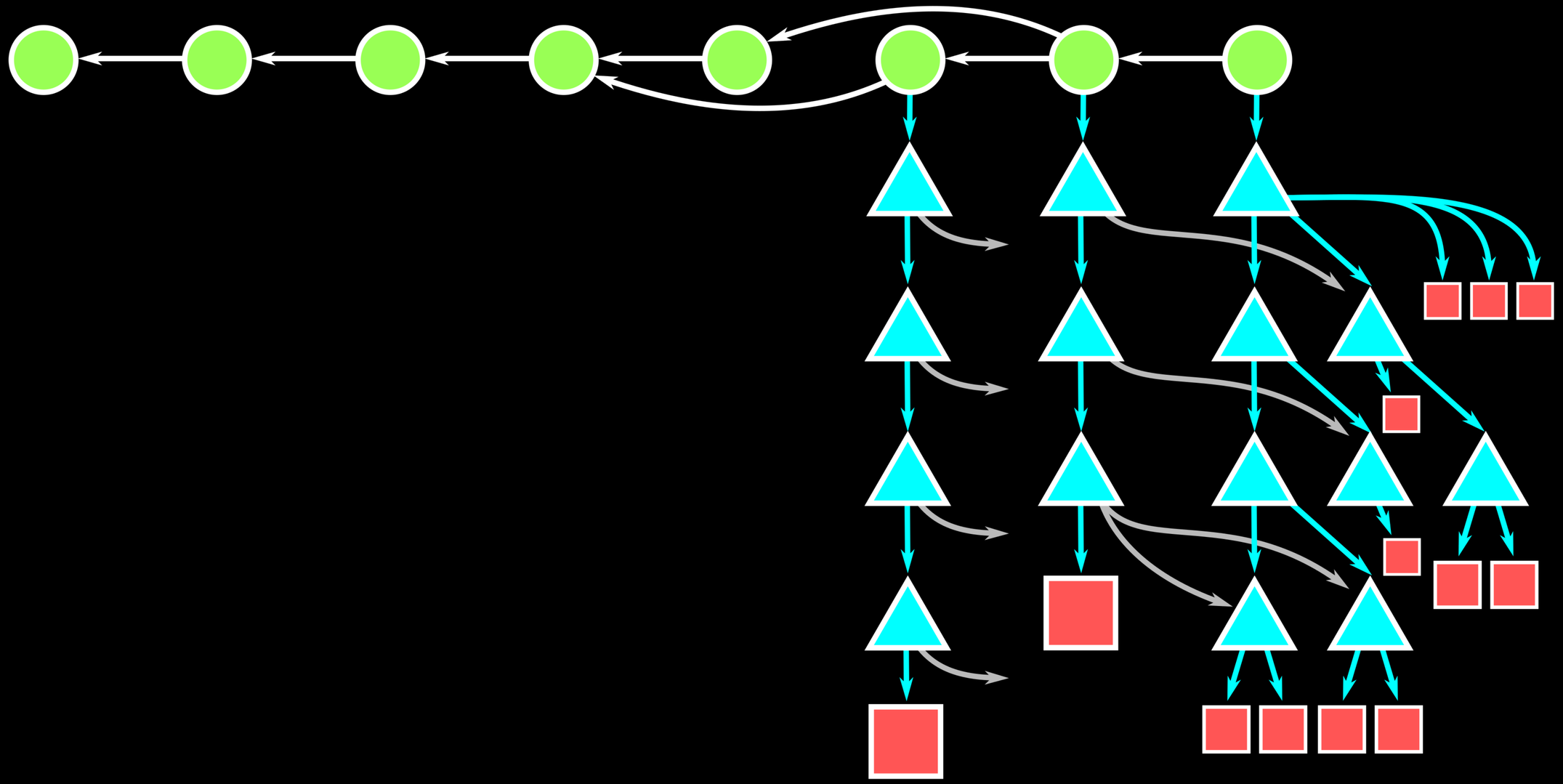

“Root tree”

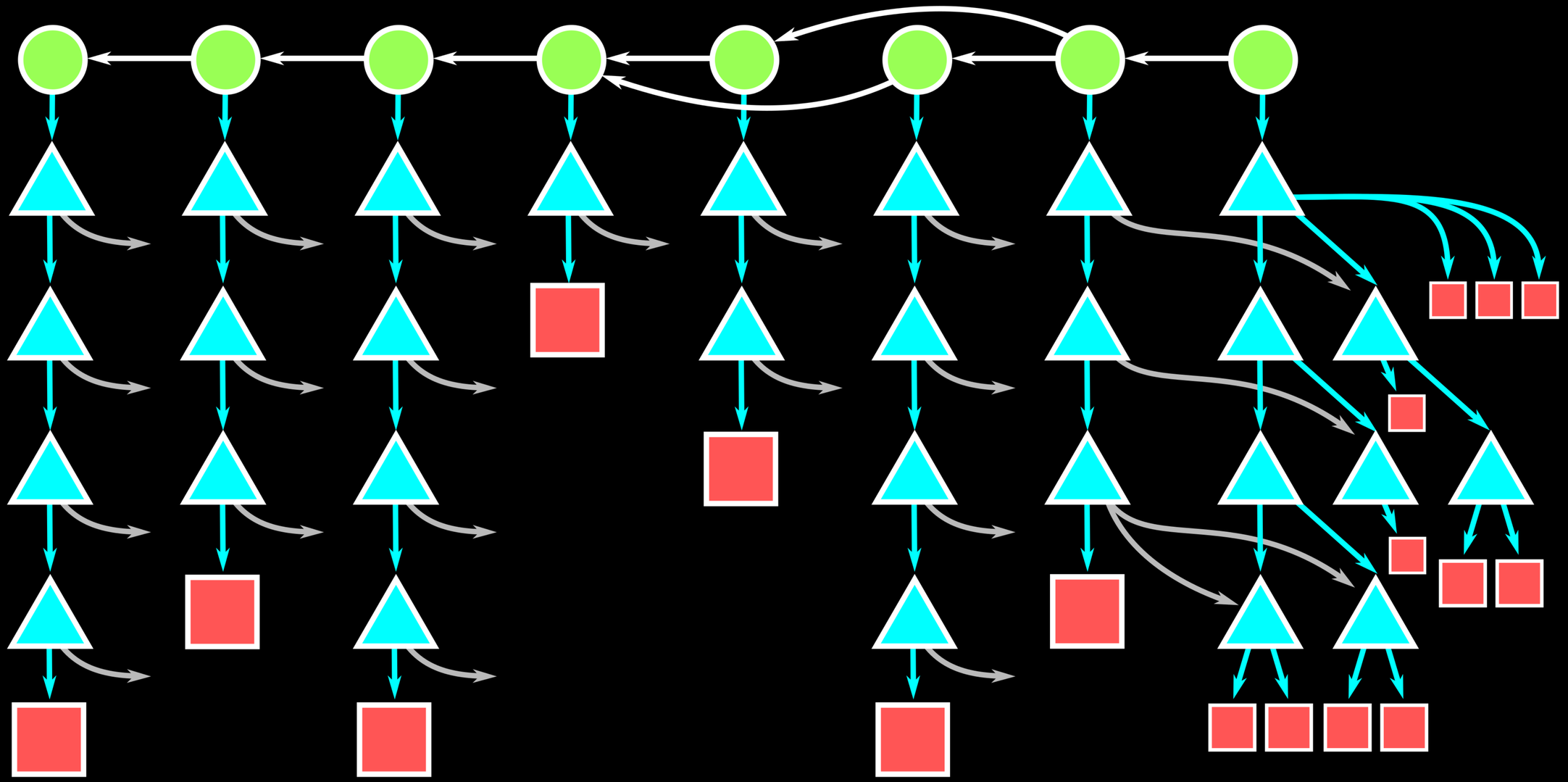

Trees
(directories)
Blobs
(files)

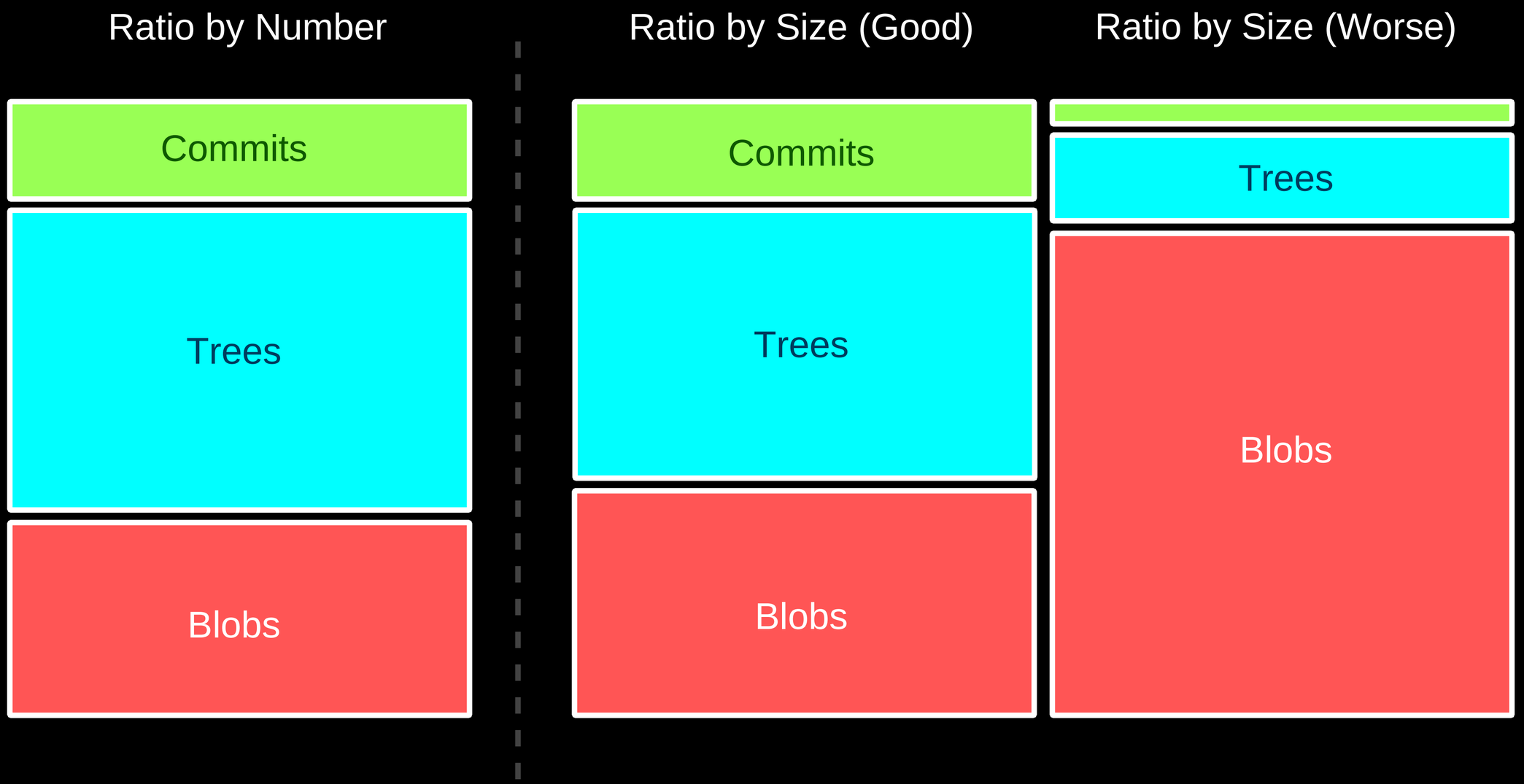




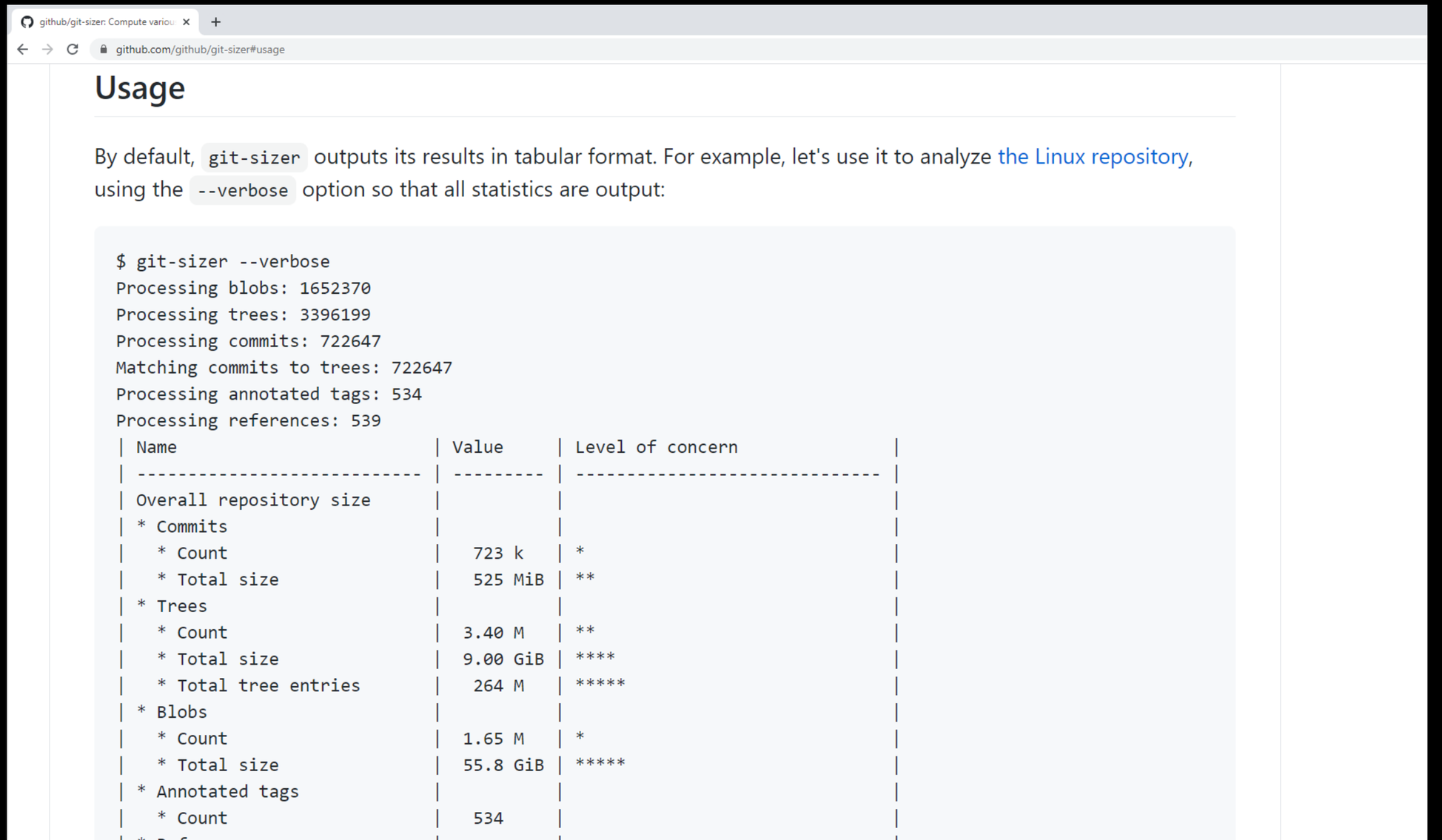







Ratio by Number
Commits
Trees
Blobs
Ratio by Size (Good)
Commits
Trees
Blobs
Ratio by Size (Worse)
Trees
Blobs

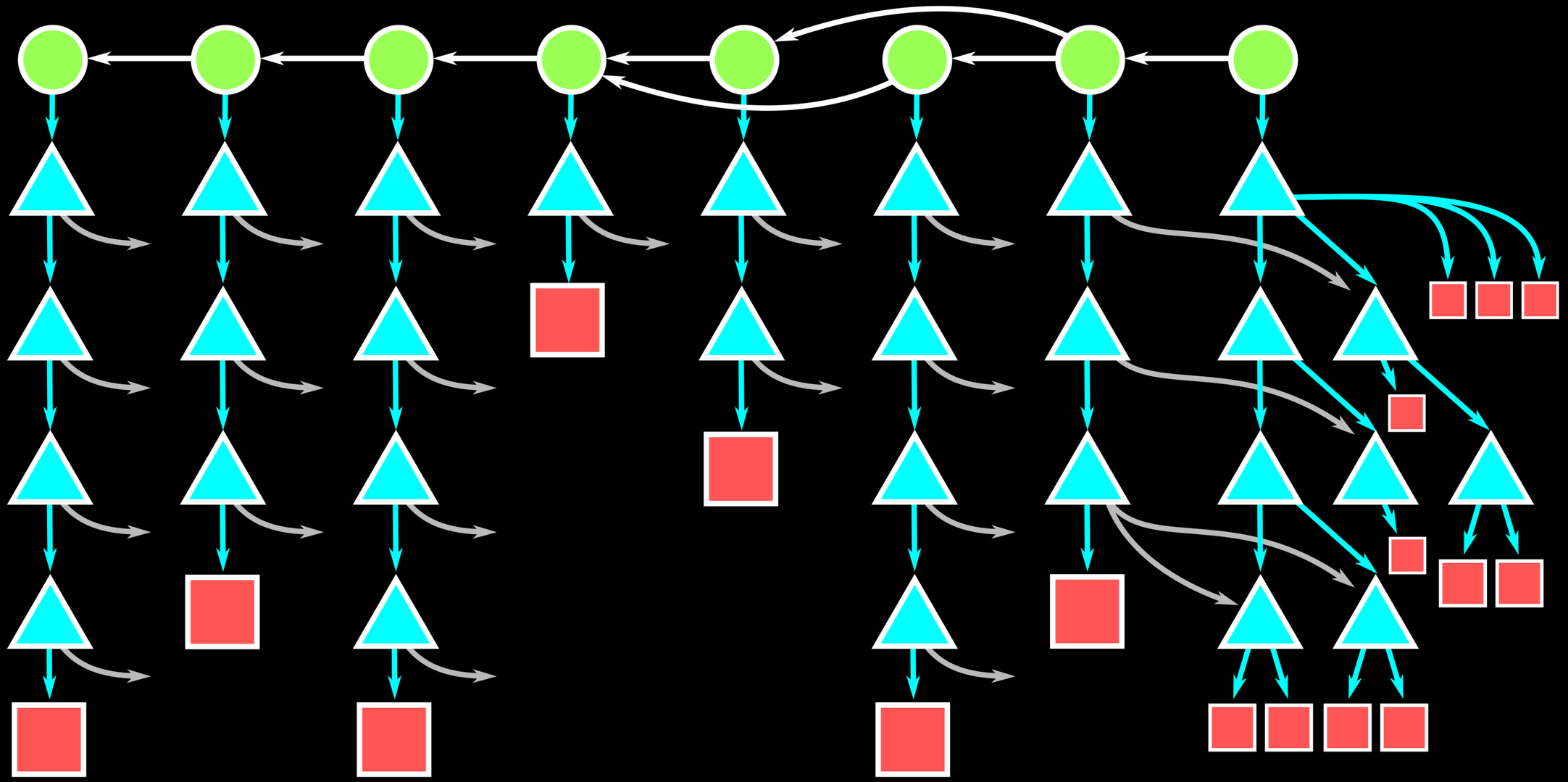
https://github.com/github/git-sizer



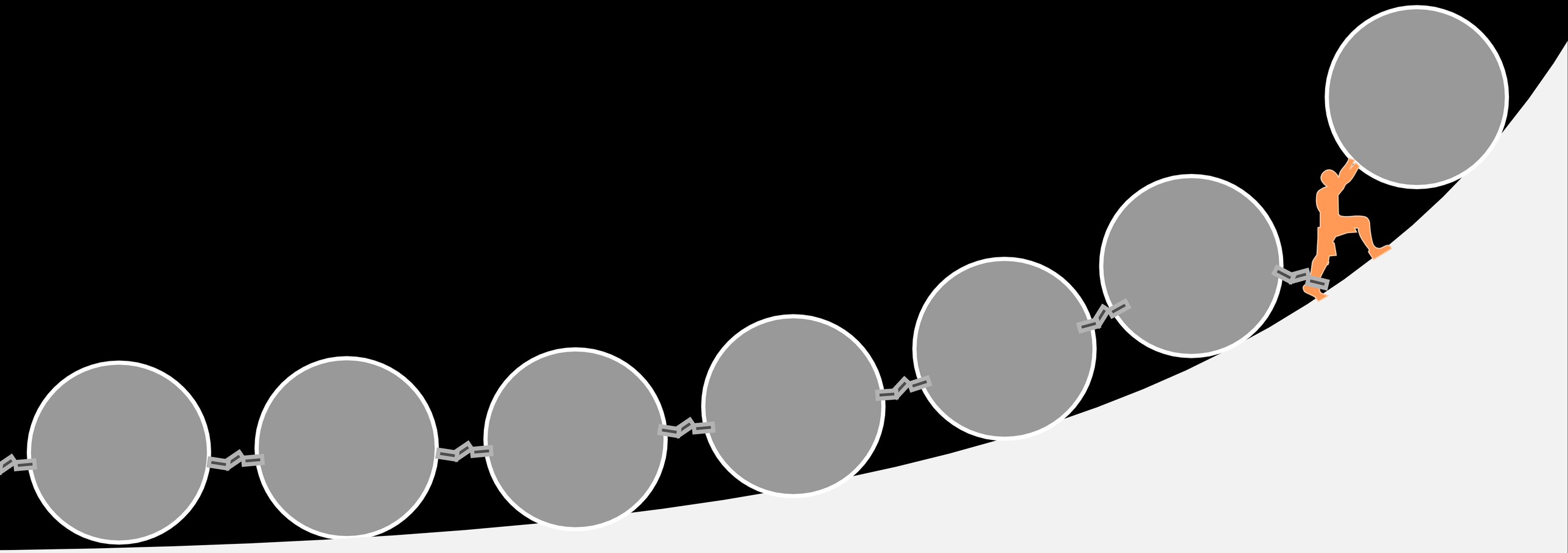
Usage

By default, `git-sizer` outputs its results in tabular format. For example, let's use it to analyze the Linux repository, using the `--verbose` option so that all statistics are output:

```
$ git-sizer --verbose
Processing blobs: 1652370
Processing trees: 3396199
Processing commits: 722647
Matching commits to trees: 722647
Processing annotated tags: 534
Processing references: 539
| Name                         | Value     | Level of concern               |
| ---------------------------- | --------- | ------------------------------ |
| Overall repository size      |           |                                |
| * Commits                    |           |                                |
|   * Count                    |   723 k   | *                              |
|   * Total size               |   525 MiB | **                             |
| * Trees                      |           |                                |
|   * Count                    |  3.40 M   | **                             |
|   * Total size               |  9.00 GiB | ****                           |
|   * Total tree entries       |   264 M   | *****                          |
| * Blobs                      |           |                                |
|   * Count                    |  1.65 M   | *                              |
|   * Total size               |  55.8 GiB | *****                          |
| * Annotated tags             |           |                                |
|   * Count                    |   534     |                                |
```



Large blobs dragging you down?



Kick out bad blobs!

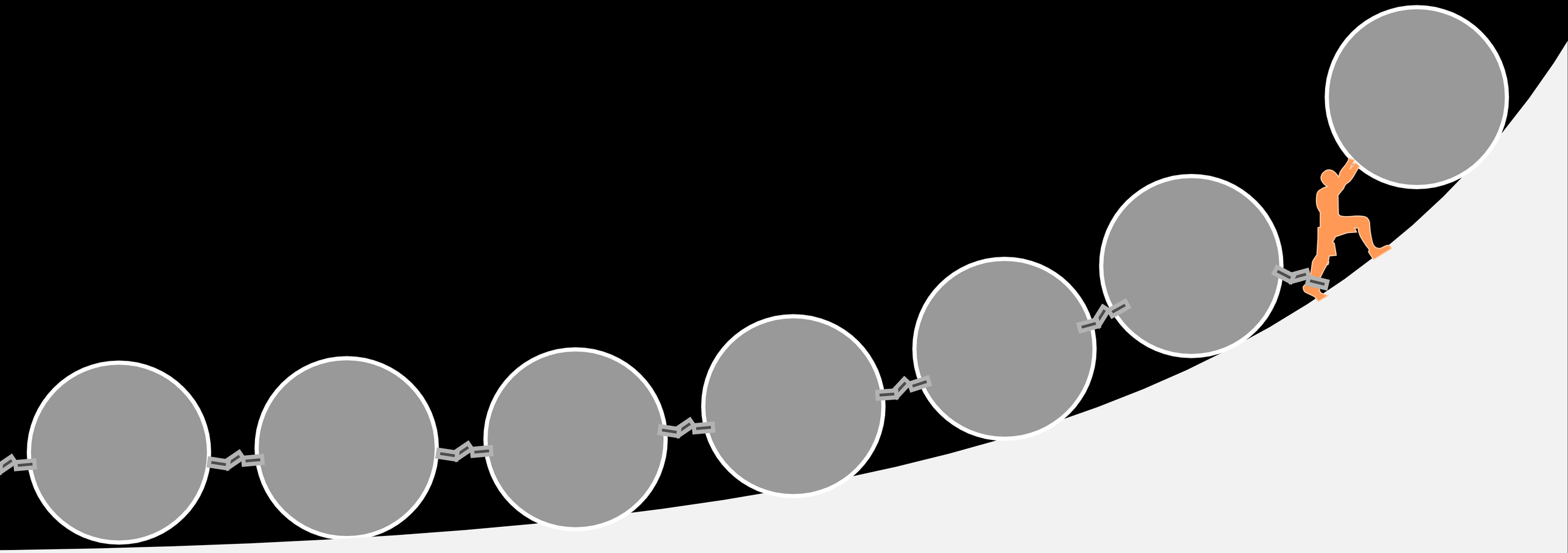
Git LFS

GitHub Packages

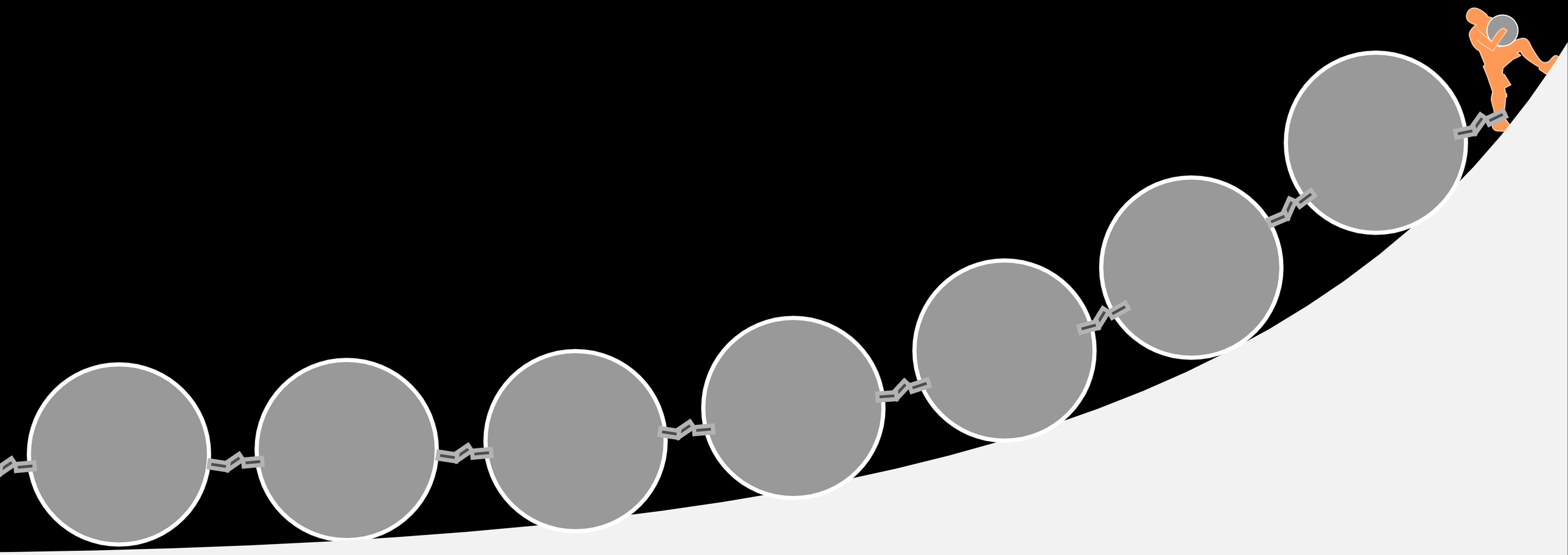
HOT TAKE

If you can’t **review it**,
then **delete it!**

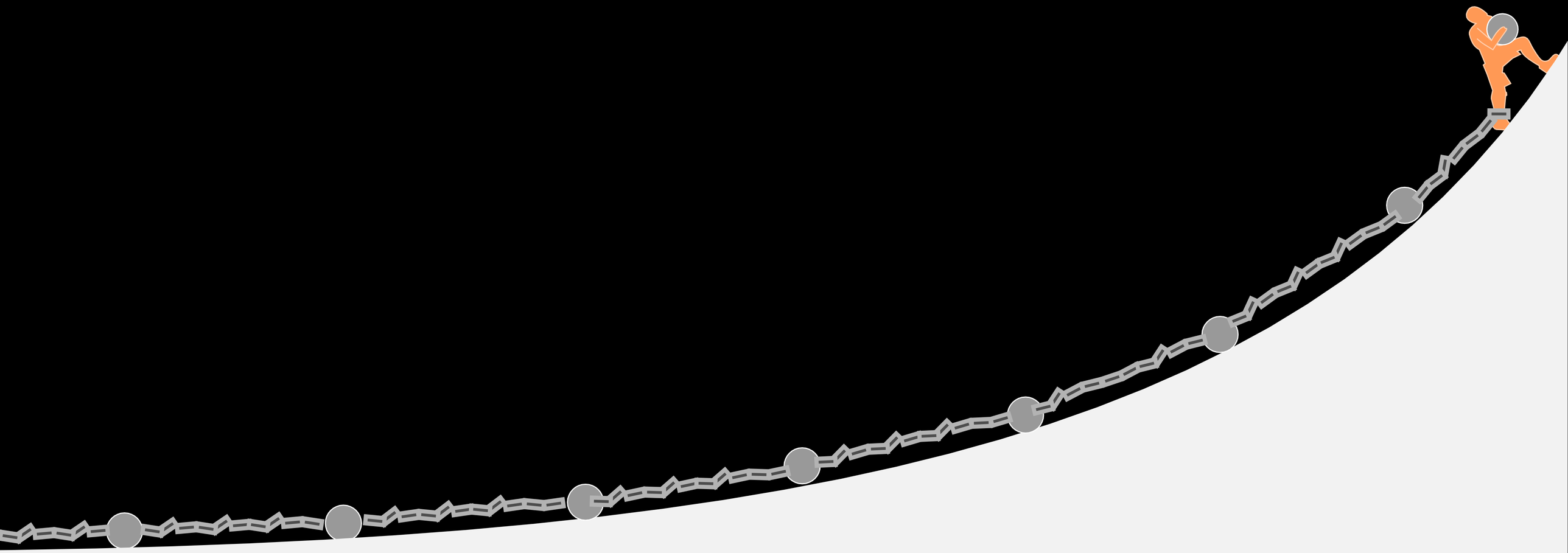
Carry that weight



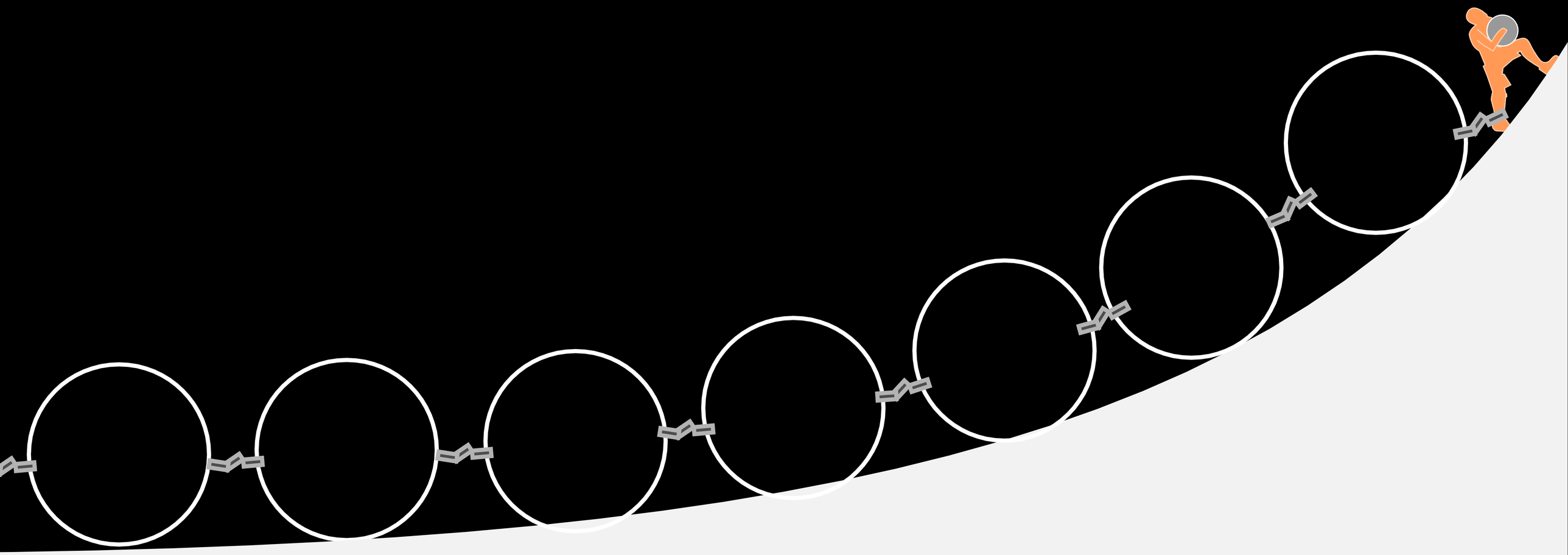
Remove files at tip



Rewrite history?



Lighten the weight

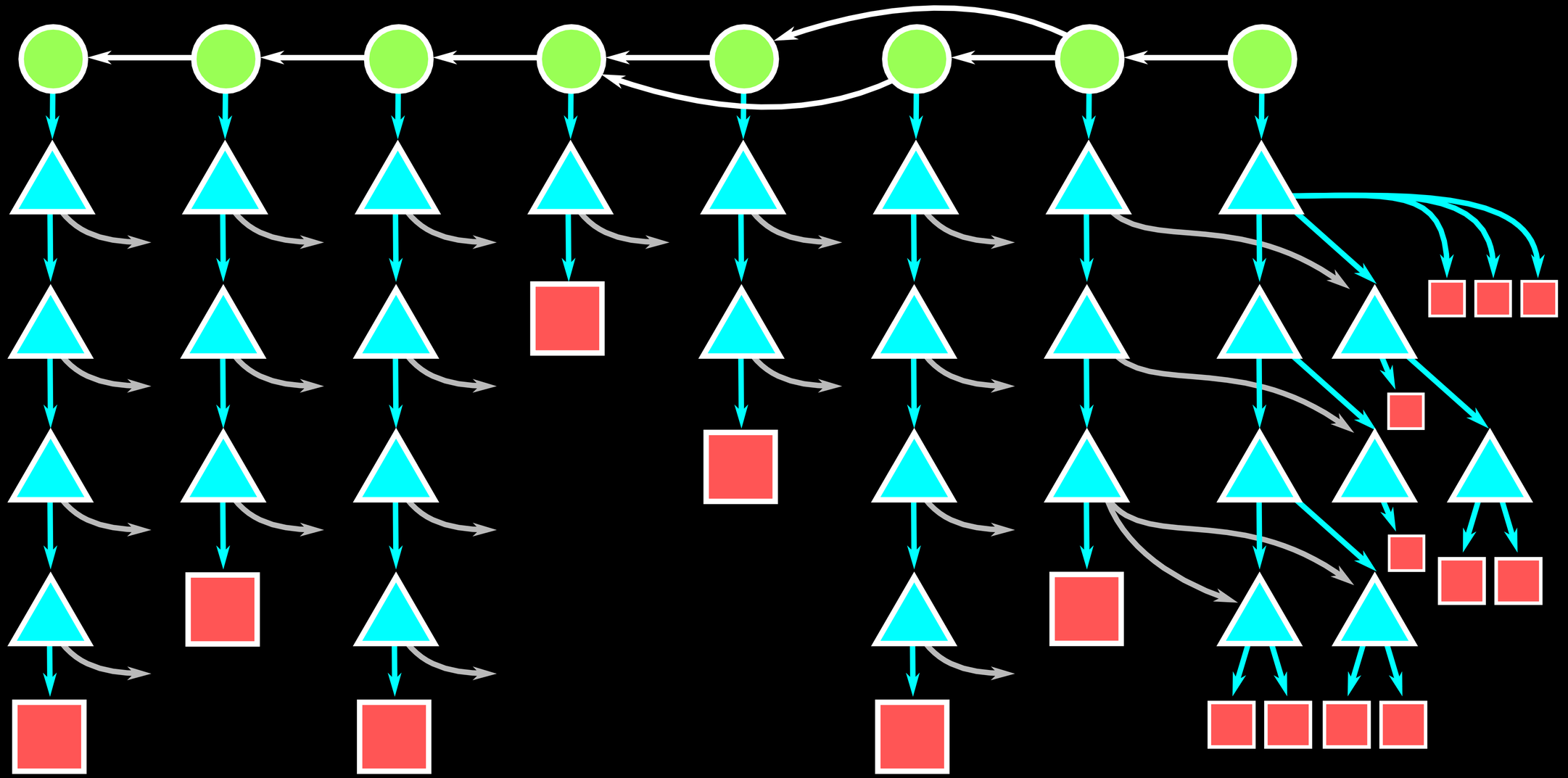


Pro Tip: Use partial clone

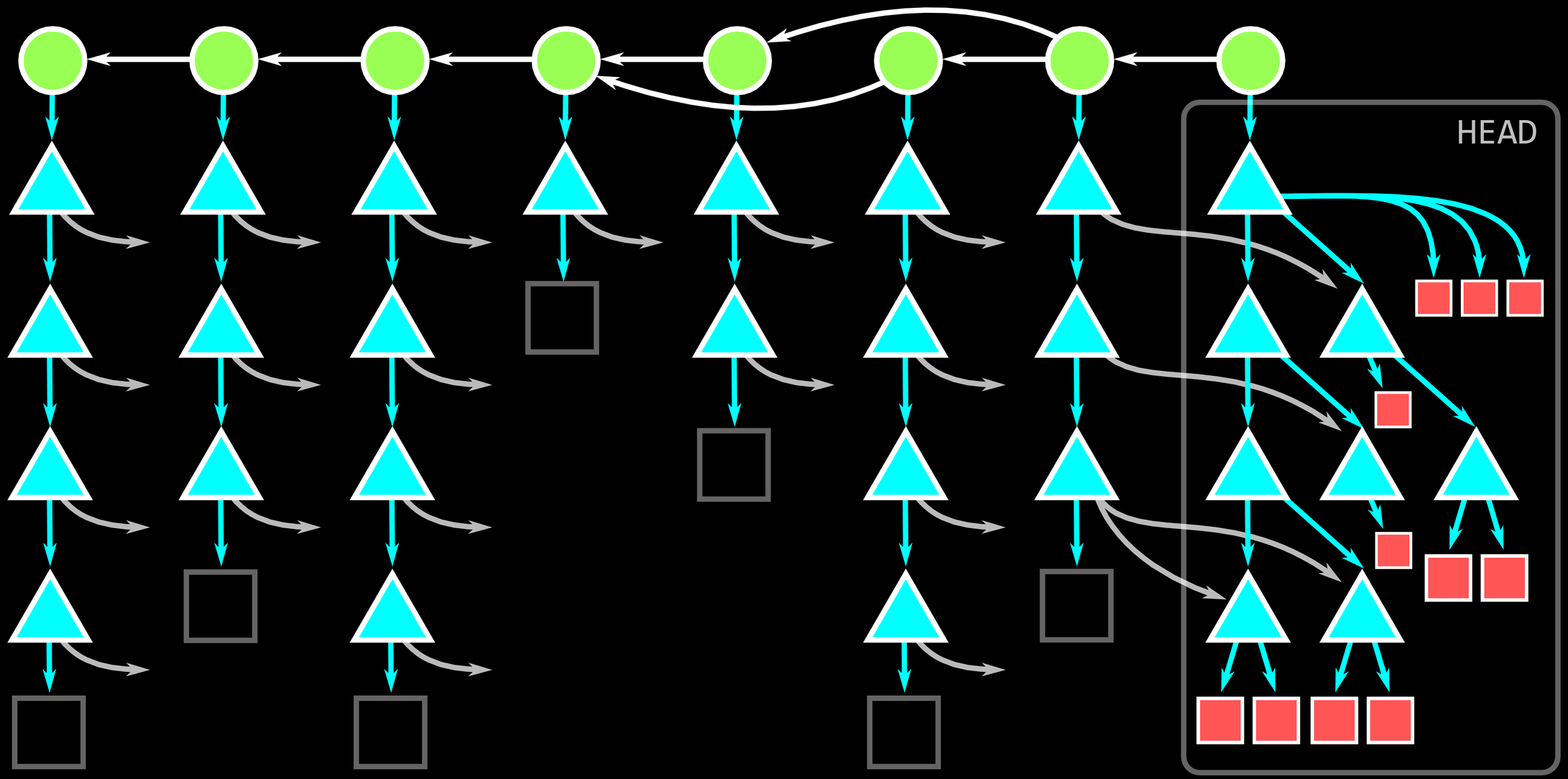
```
git clone --filter=blob:none <url>
```

Partial clones are available now!

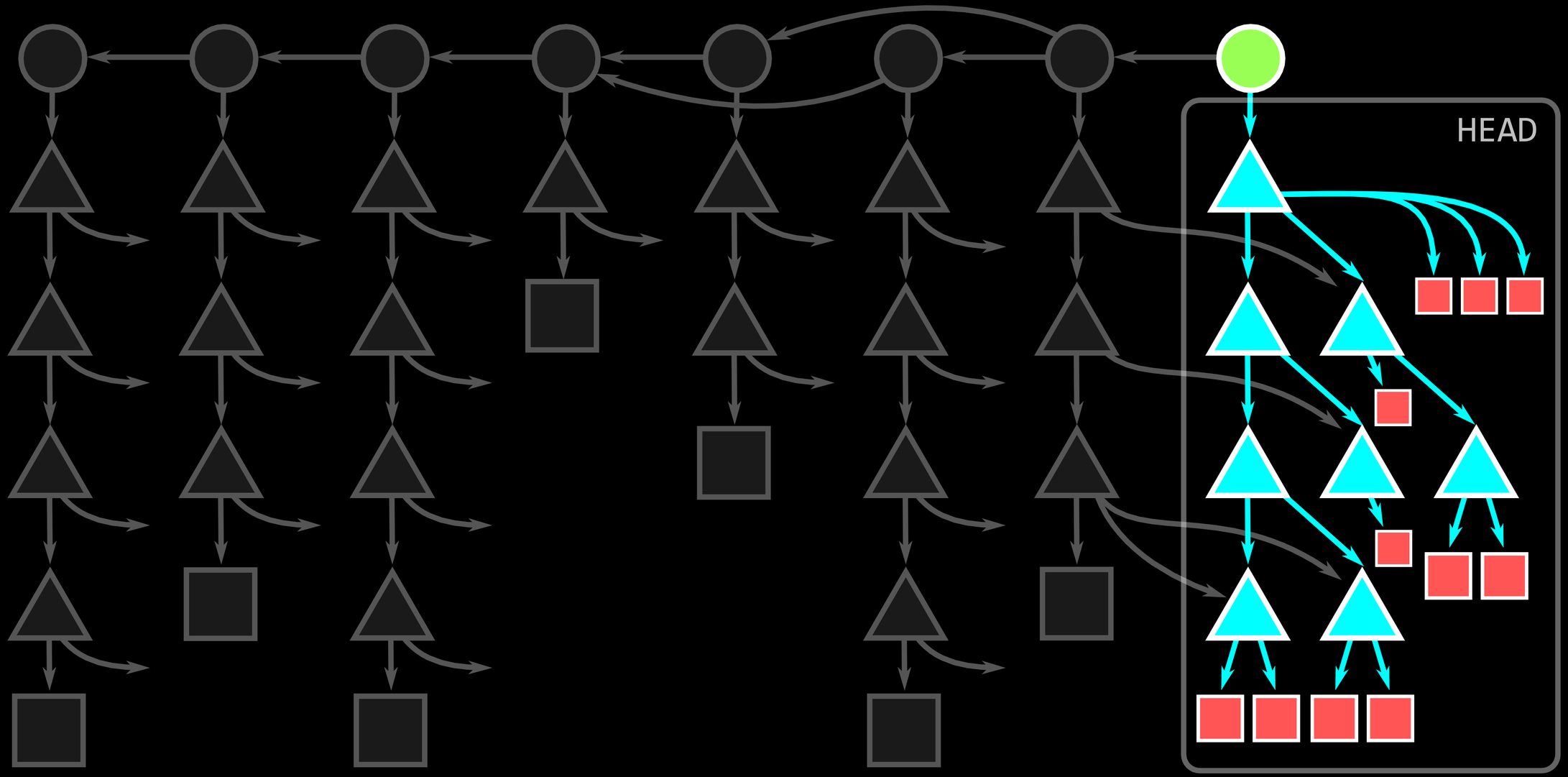
`github.com` **GHES 2.22+**



Full Clone: All commits, trees, and blobs



Partial Clone: All commits and trees; *blobs as needed*

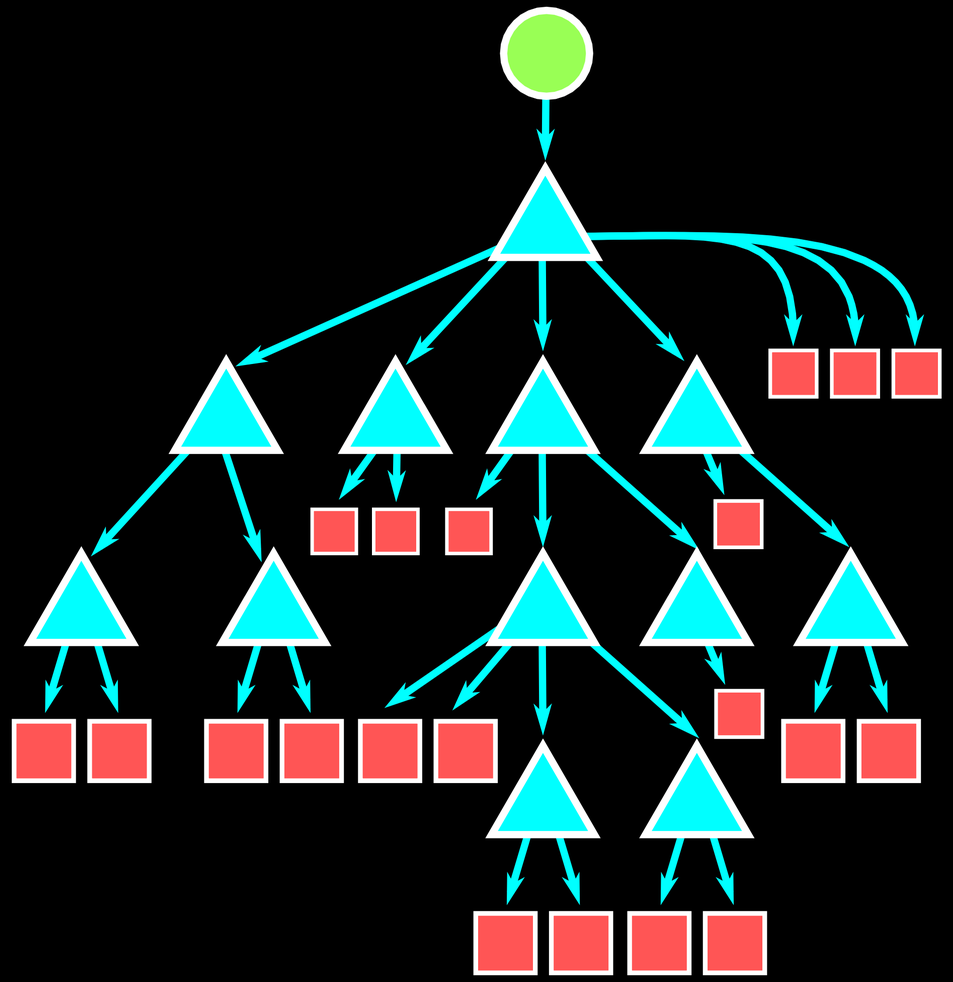


Shallow Clone: Trees and blobs for *tip commit*

HOT TAKE

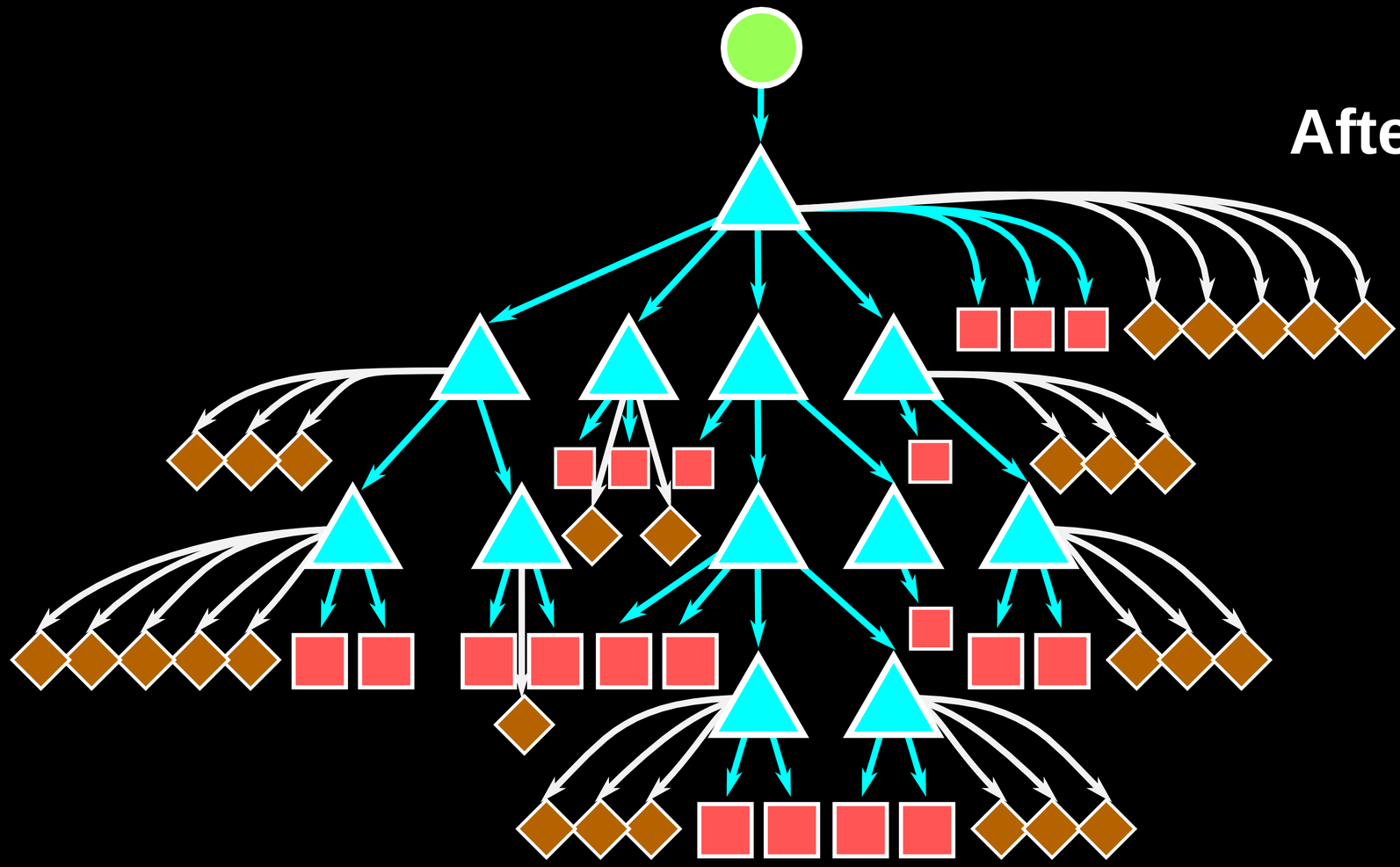
Shallow clones should be **thrown away!** 

Optimize the Working Directory

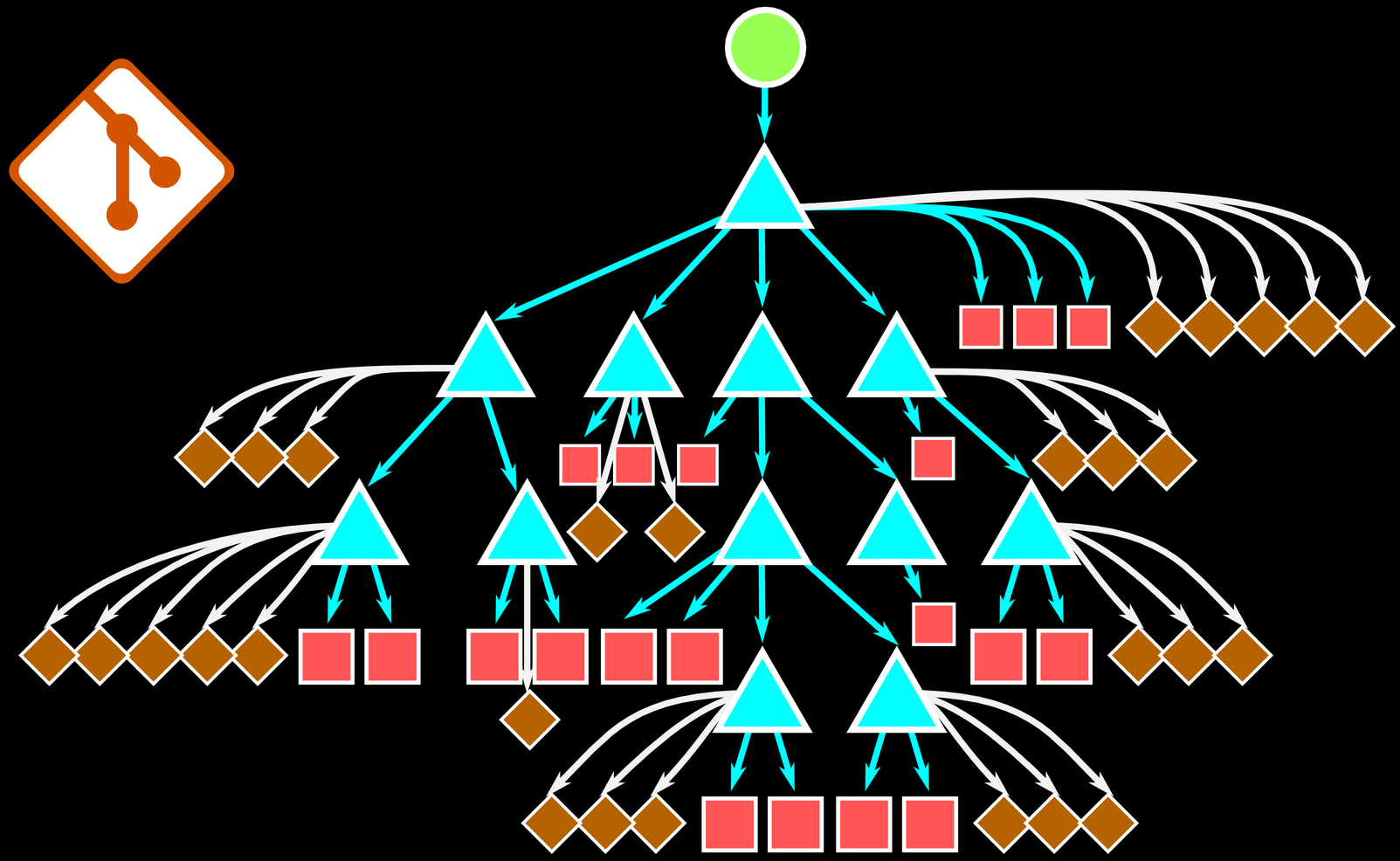


Before build

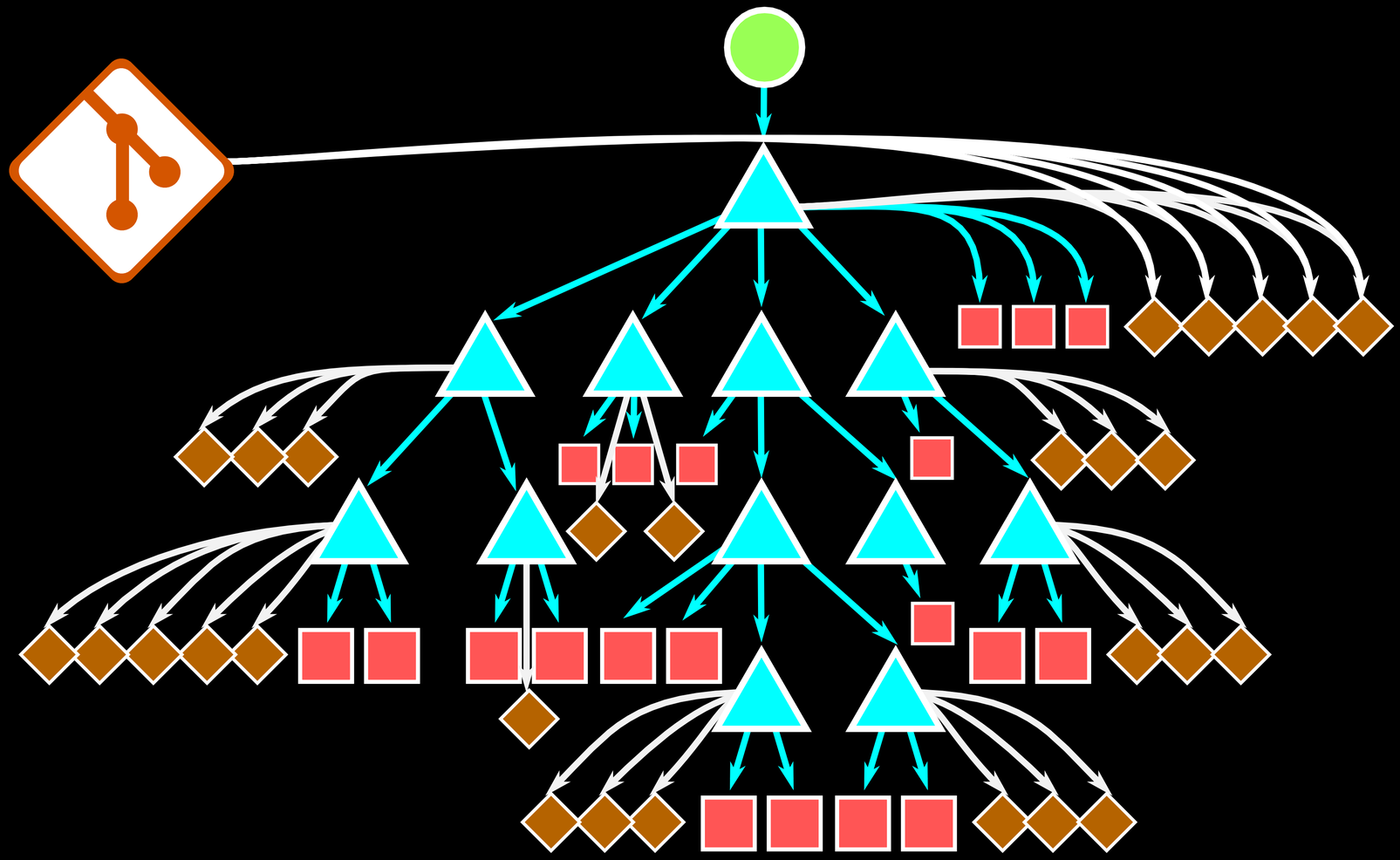
Optimize the Working Directory



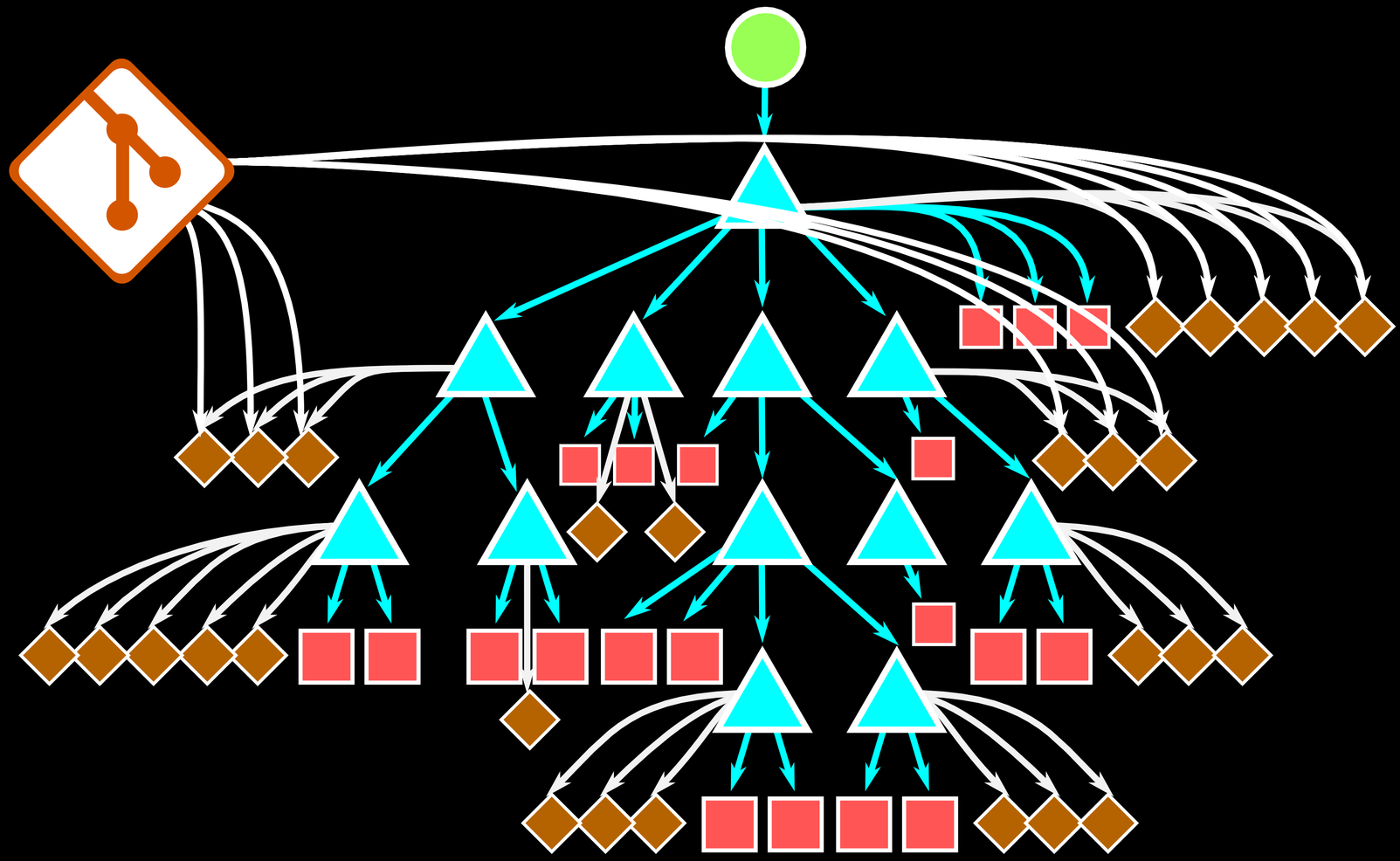
Optimize the Working Directory



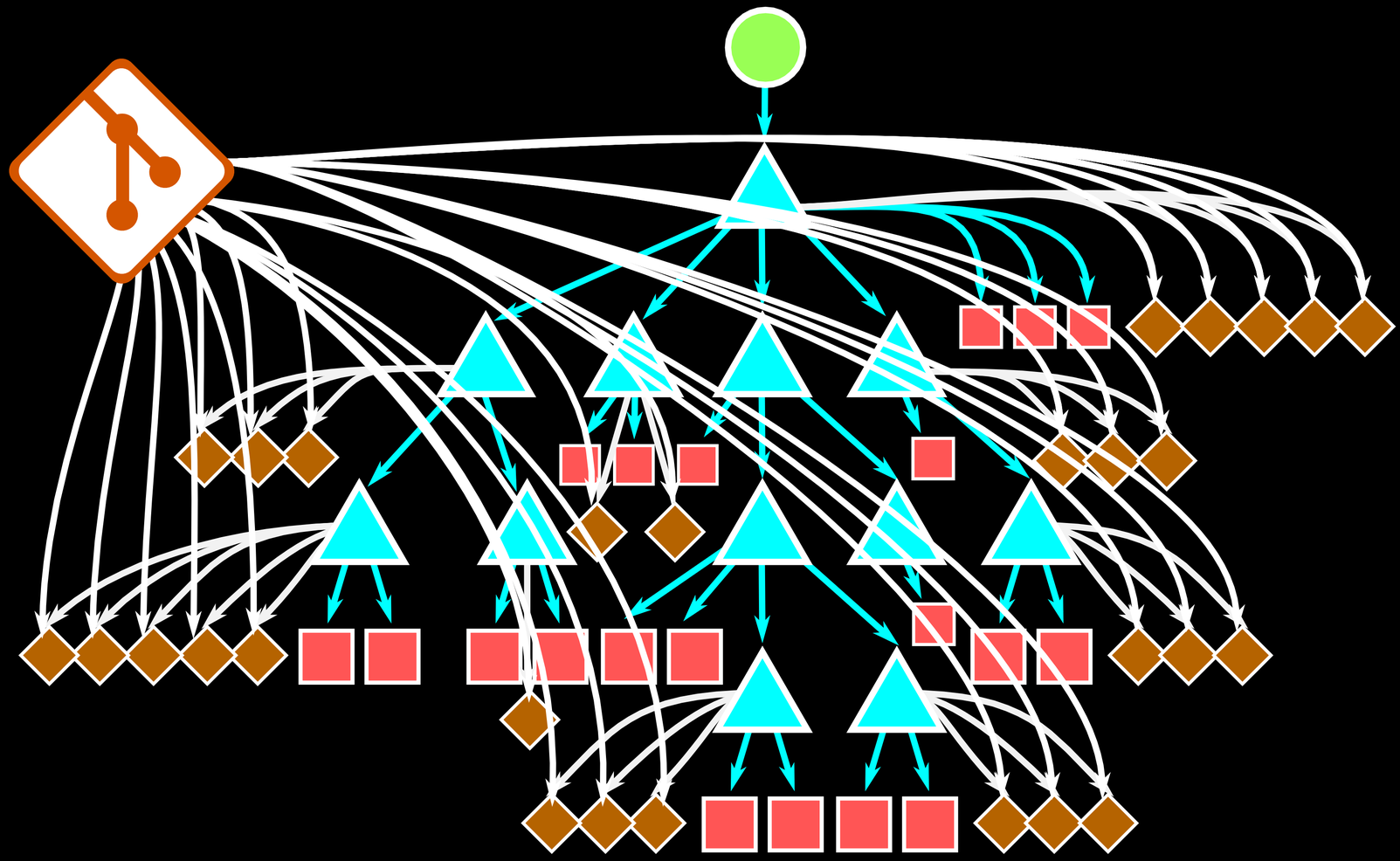
Optimize the Working Directory



Optimize the Working Directory



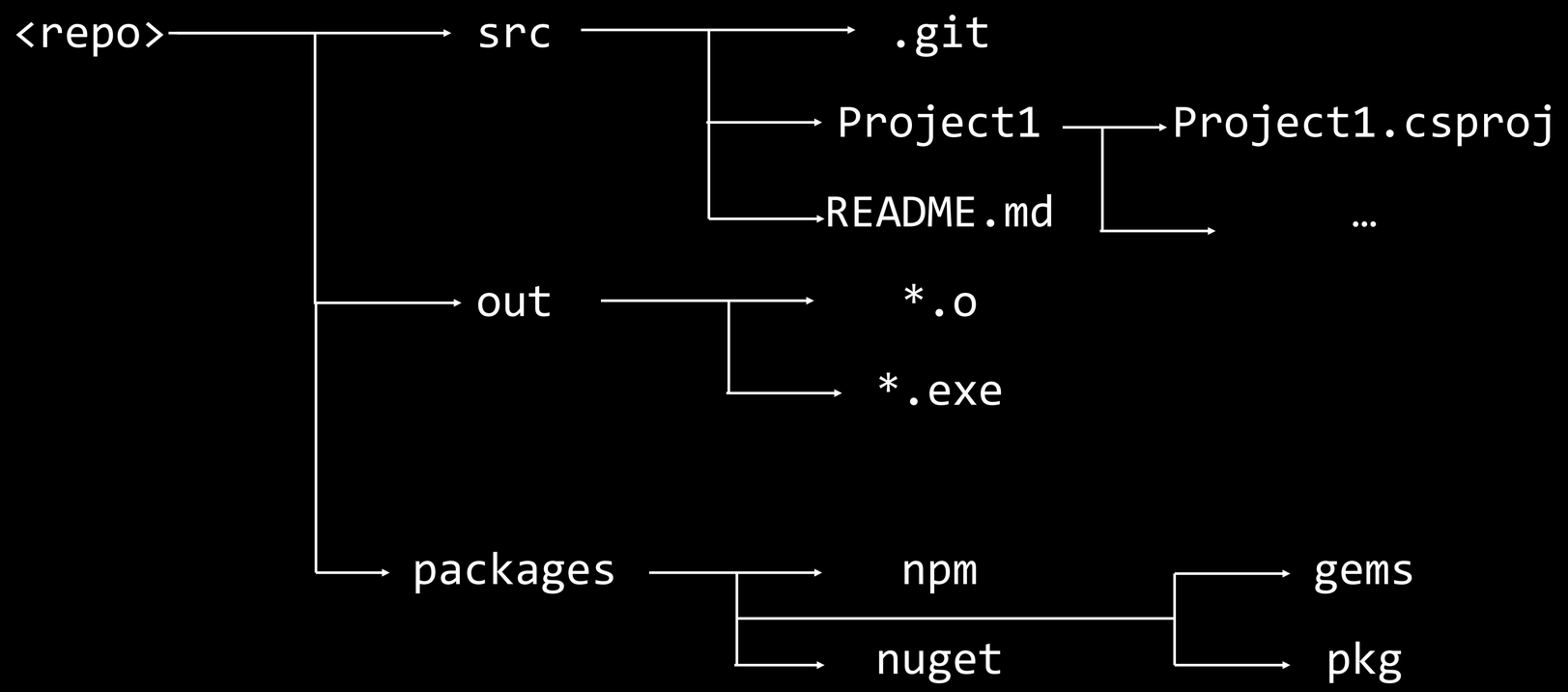
Optimize the Working Directory



HOT TAKE

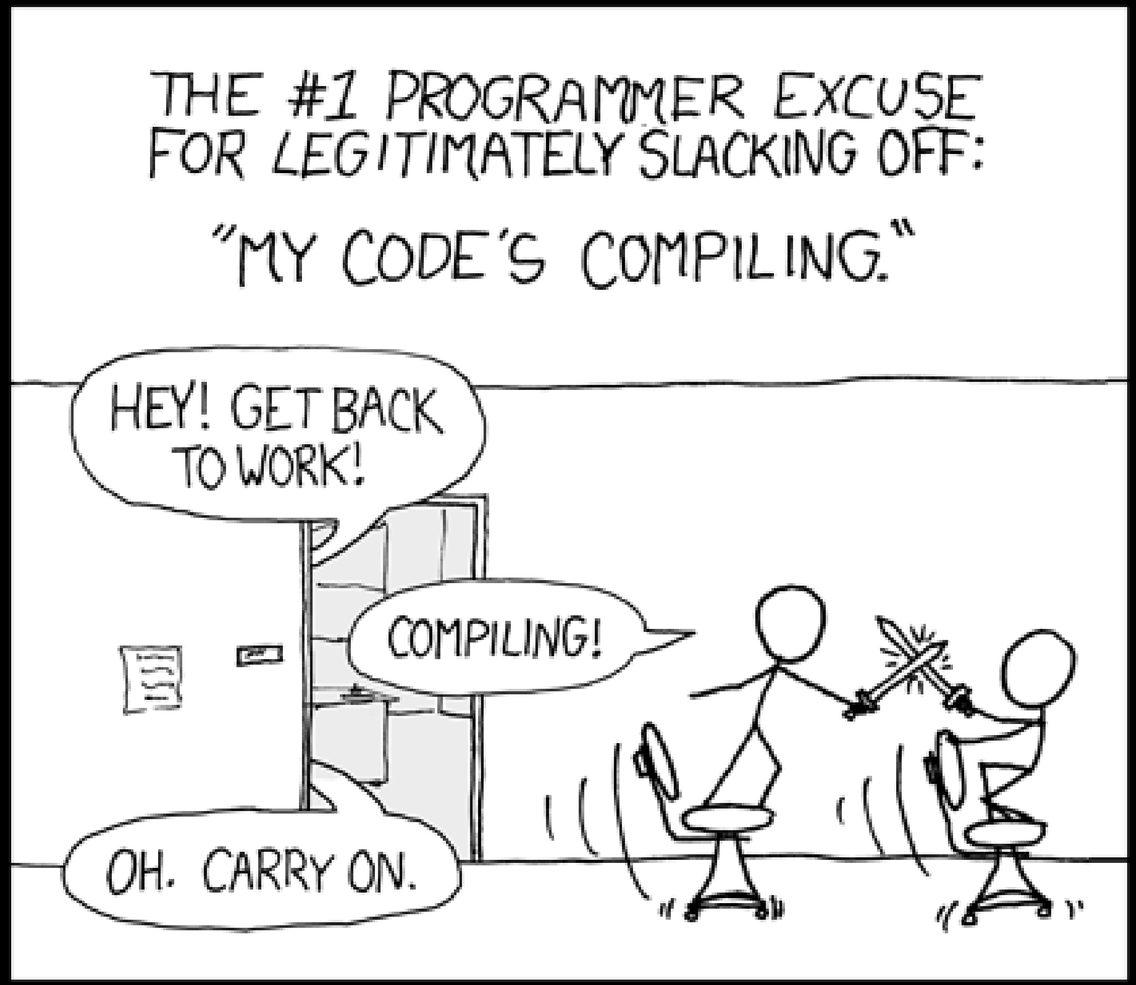
`.gitignore` files
should be tiny!

Get your builds out of src!



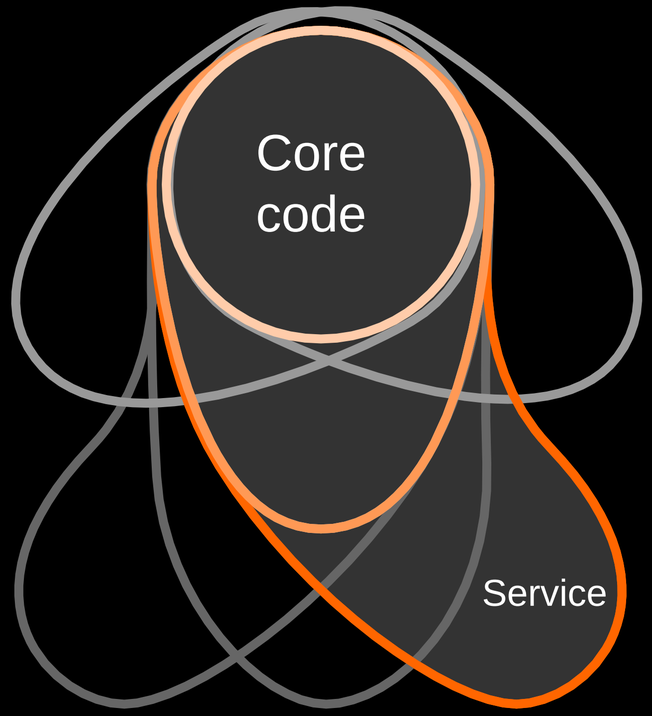
Use: `git clone <url> <repo>/src`

1. Focused builds
2. Focused devs
3. Profit!

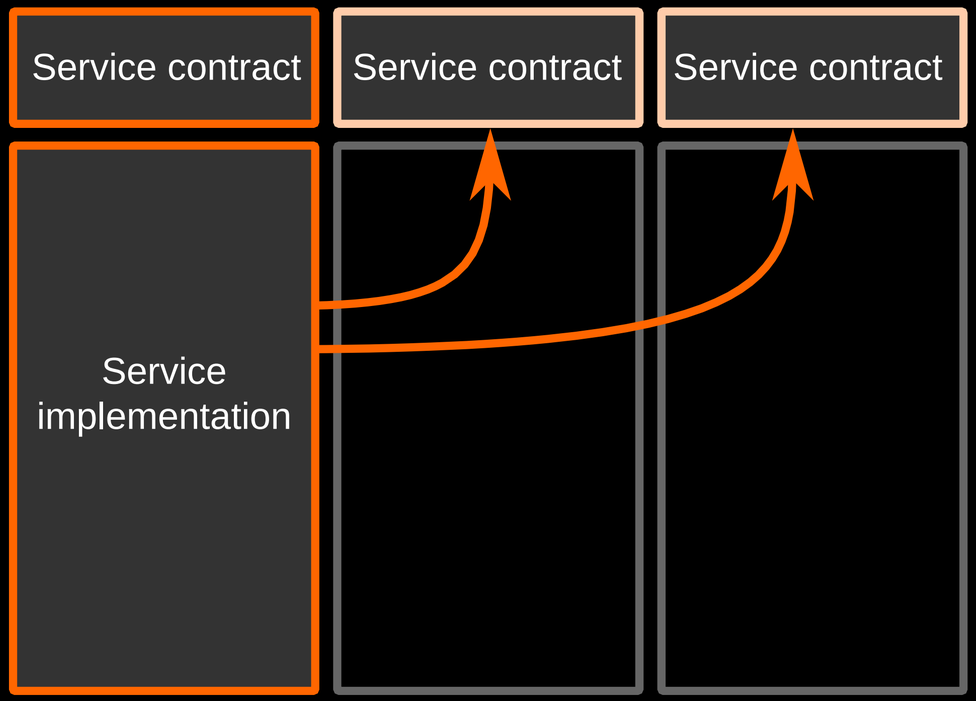


https://xkcd.com/303/

Architectures for build cones

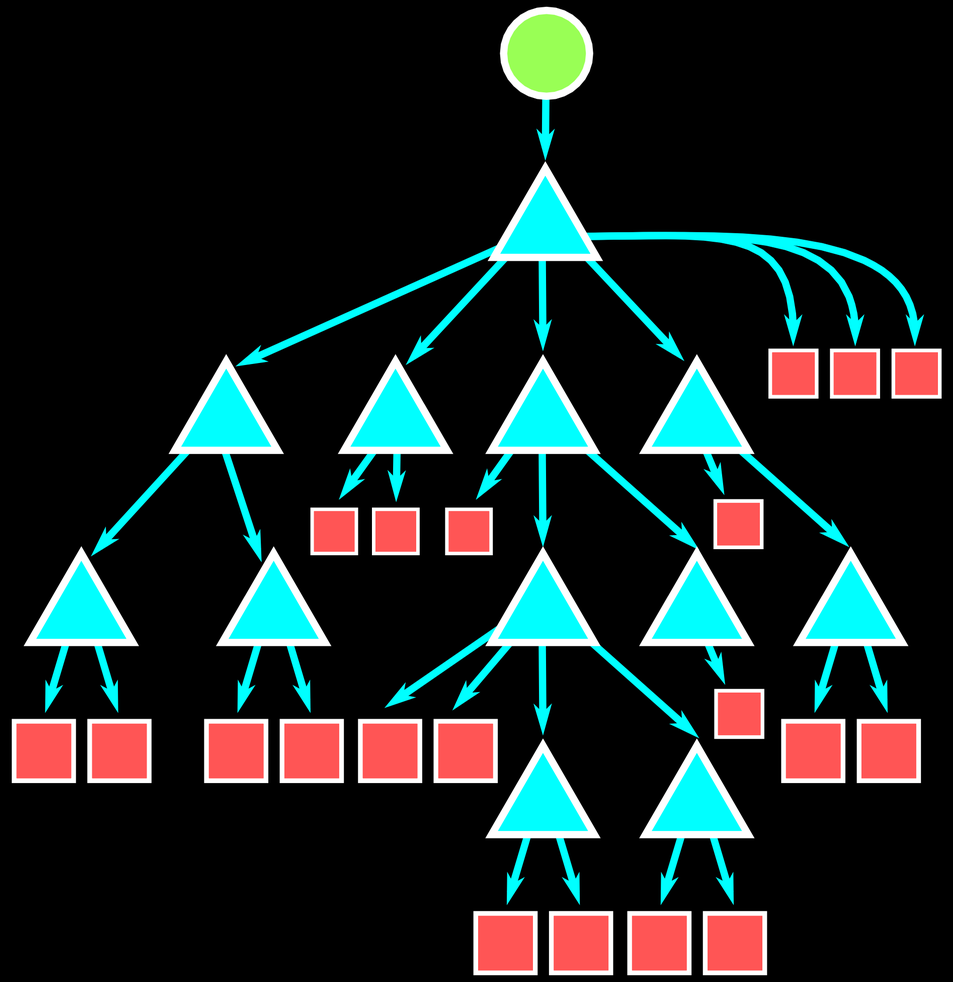


Flywheel architecture



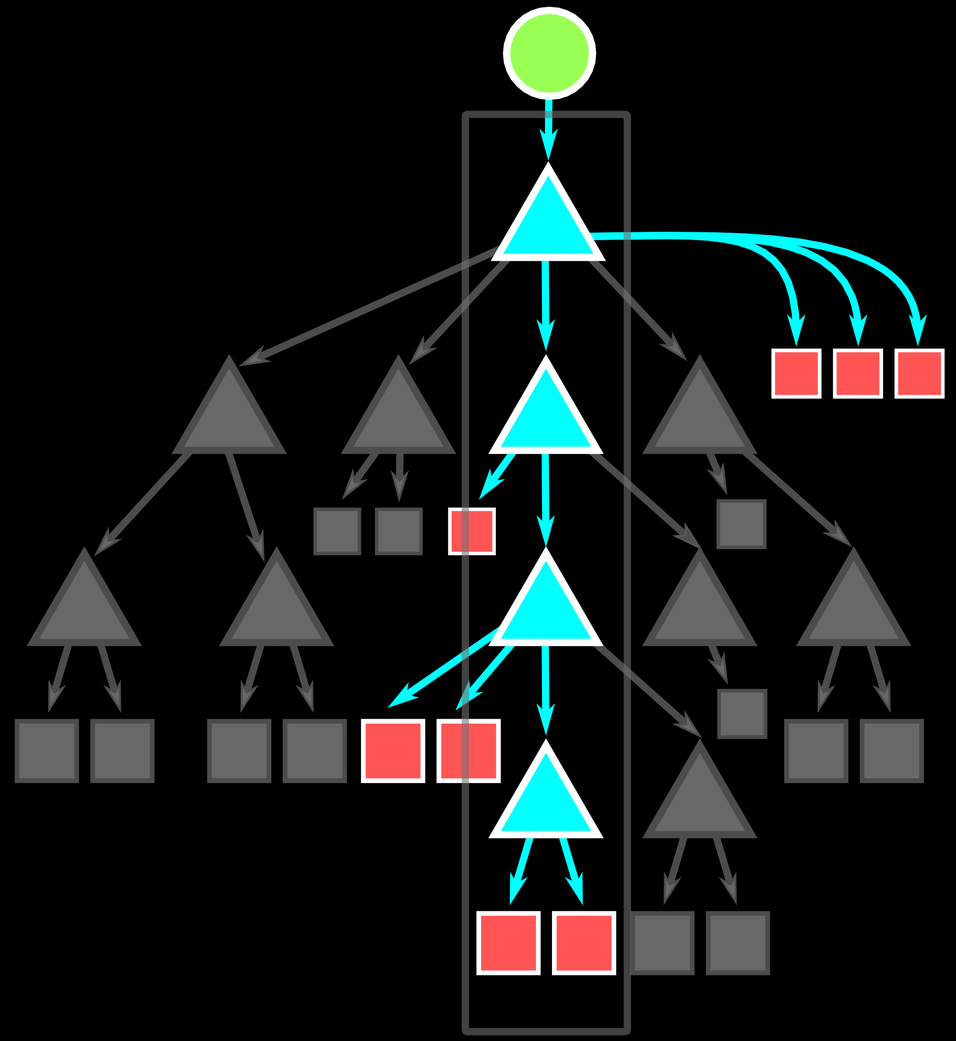
Contract-Implementation boundaries

Optimize the Working Directory



Full tree

Optimize the Working Directory

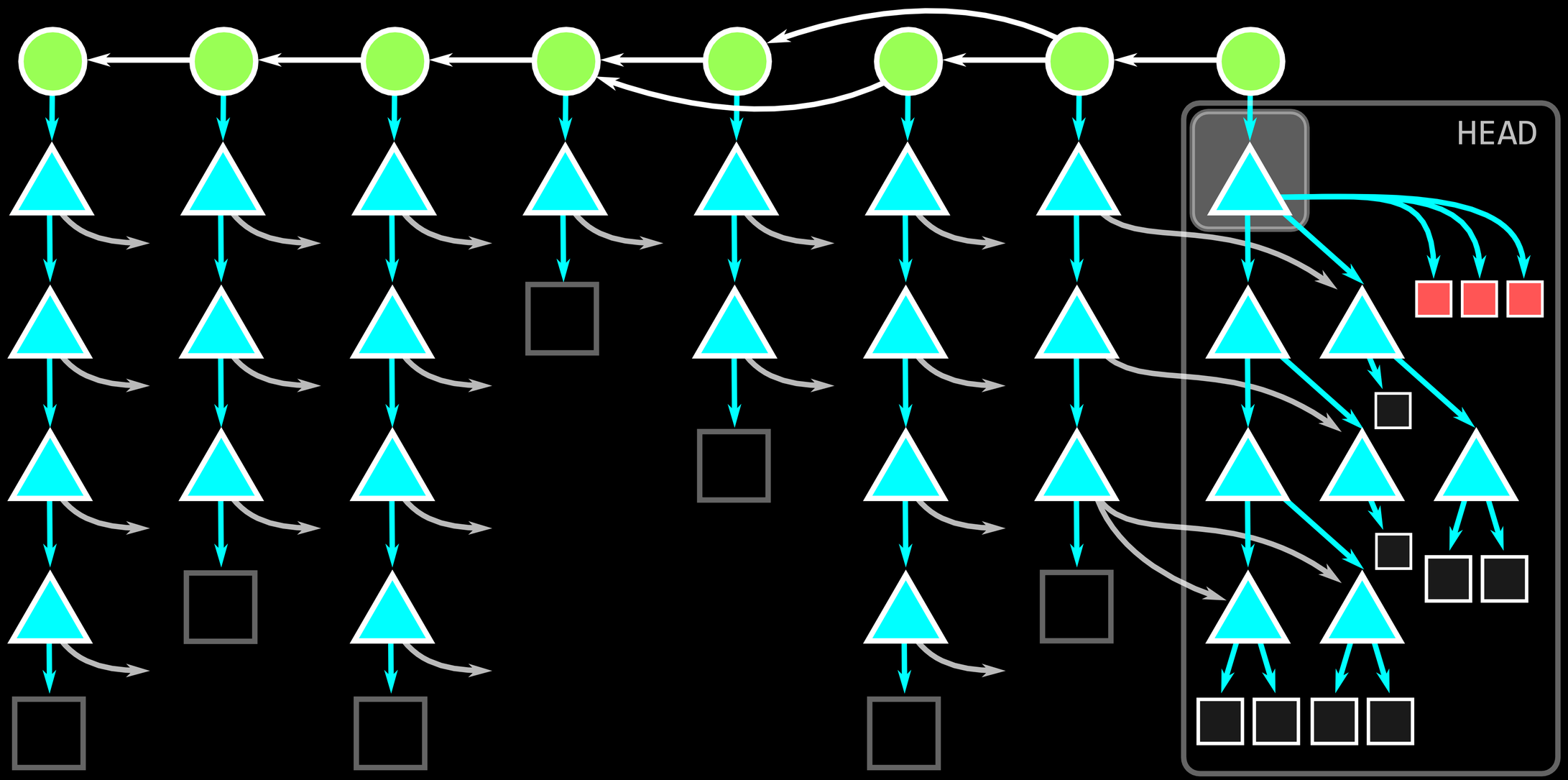


Sparse tree

Git sparse-checkout usage

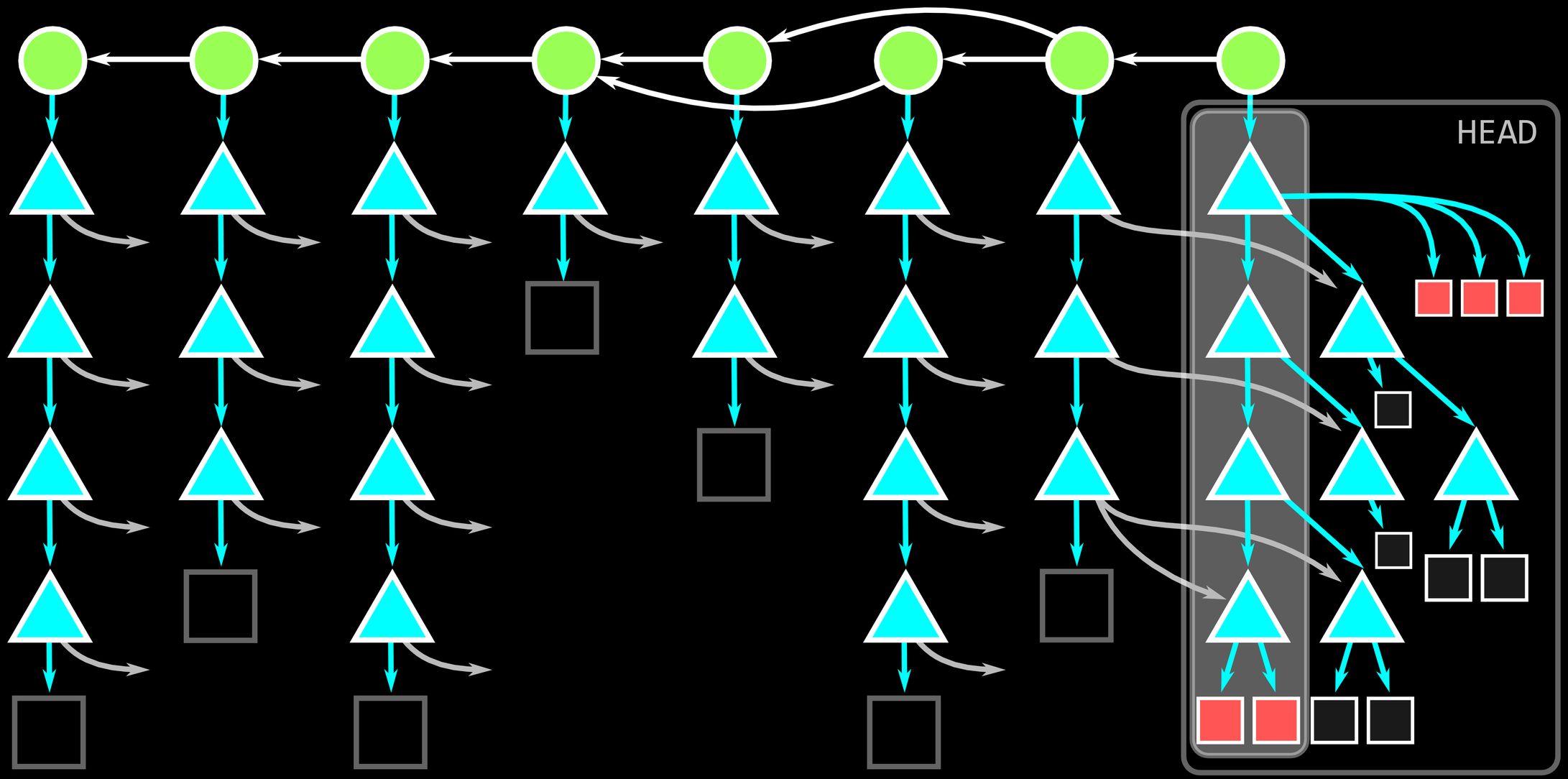
```
git sparse-checkout init --cone

git sparse-checkout set <dir1> … <dirN>

git sparse-checkout add <dir>

git sparse-checkout disable
```

https://github.blog/2020-01-17-bring-your-monorepo-down-to-size-with-sparse-checkout/



Partial Clone with Sparse-Checkout

```
git clone --filter=blob:none --sparse <url> <repo>/src
```

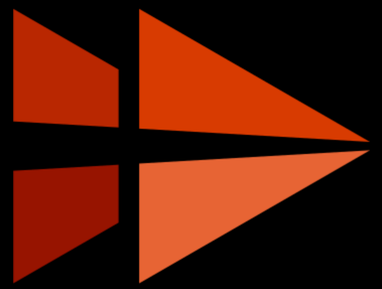


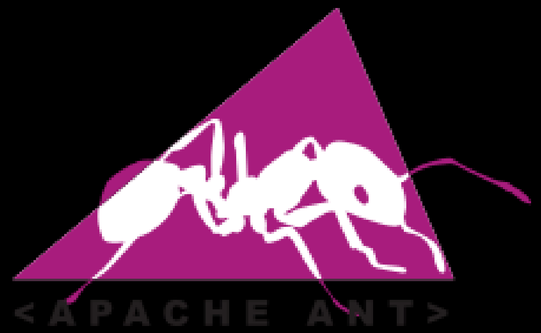
Partial Clone with Sparse-Checkout

```
git sparse-checkout add dirA/dirB/dirC
```

Call to Action

YOU are an expert in **YOUR** build system!

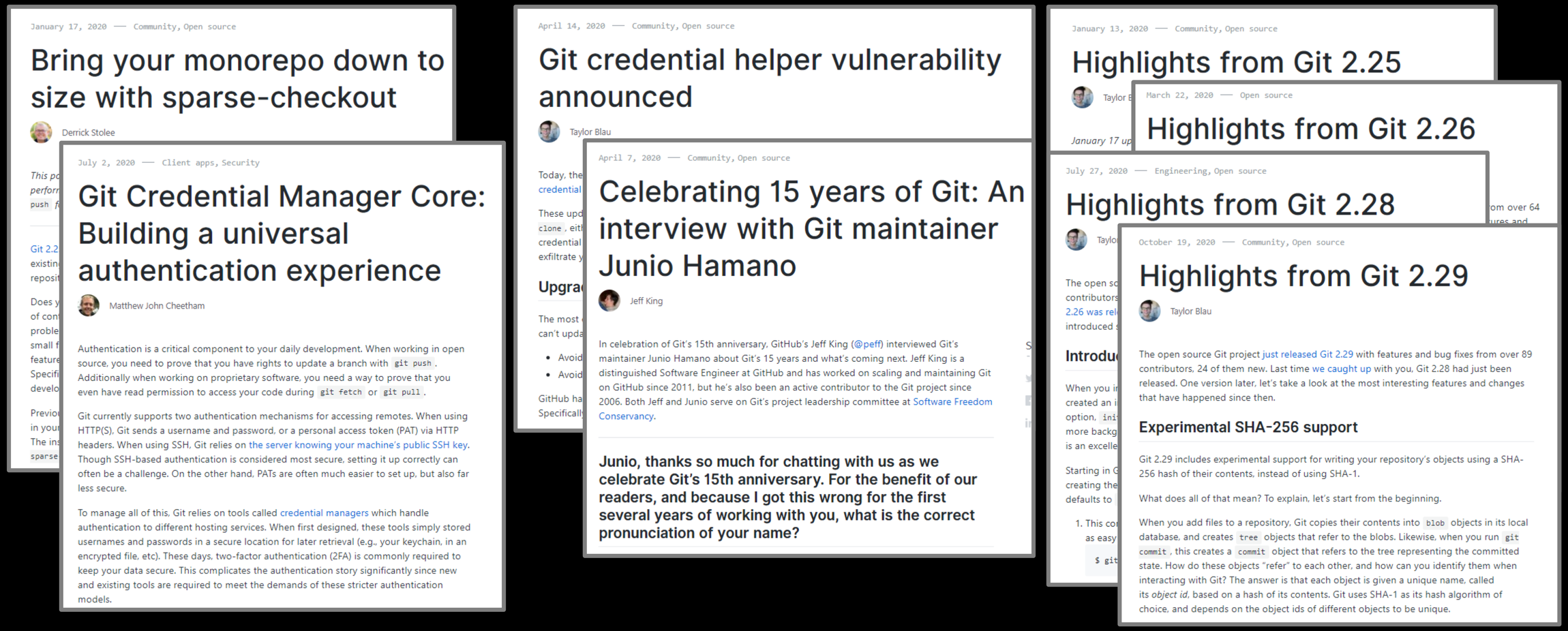


Integrate builds with `sparse-checkout`!

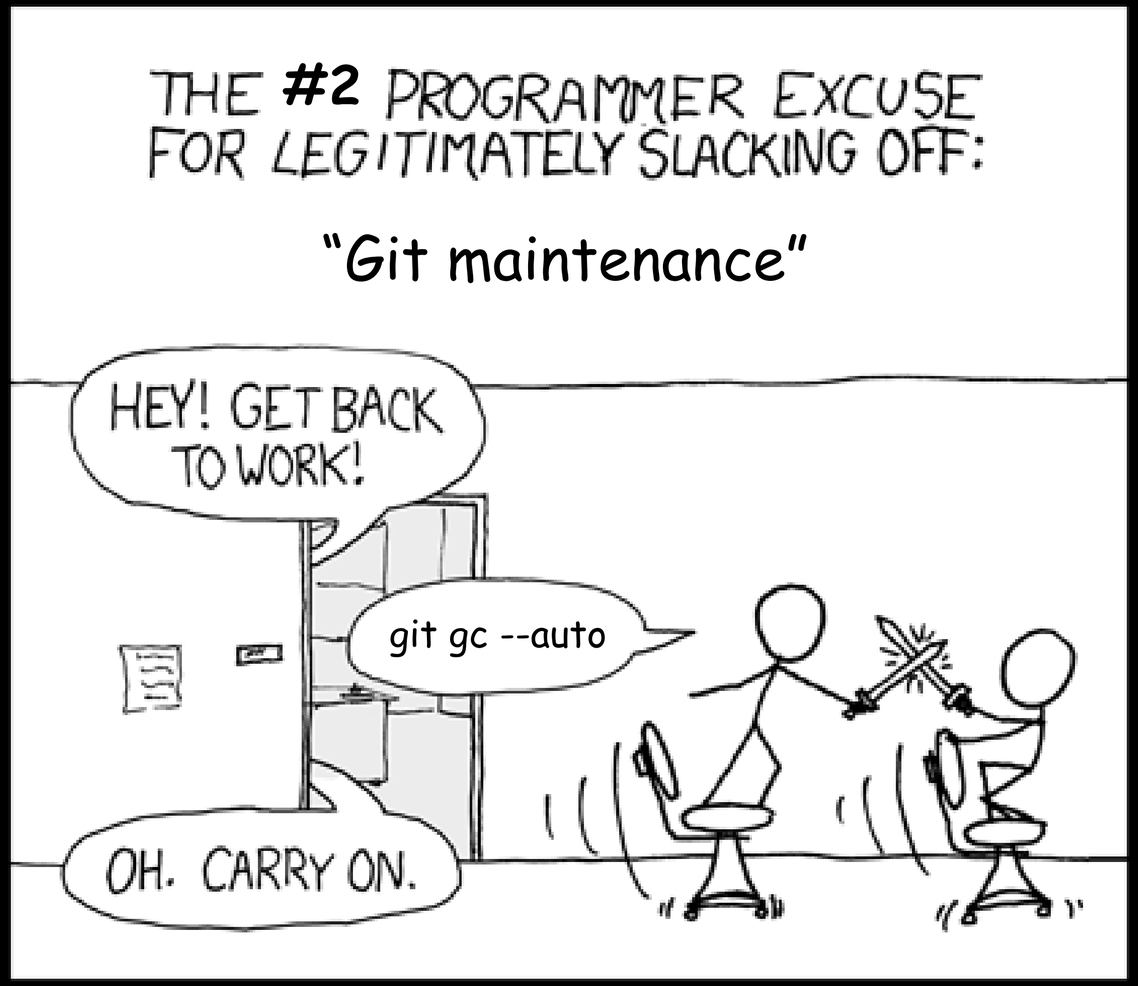
If you use these features, then you are at
the forefront of Git at scale!

Know what's up

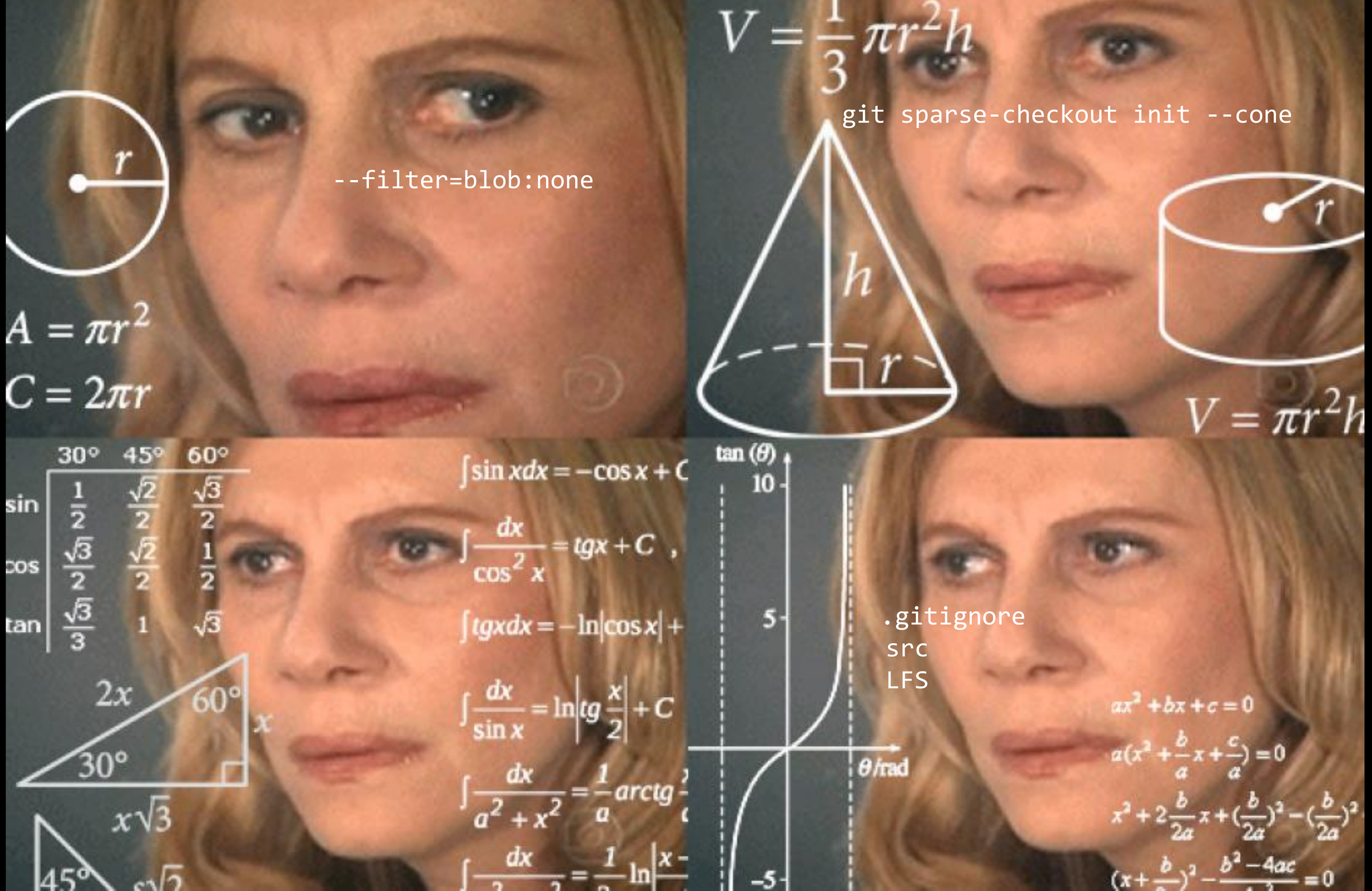


https://github.blog/

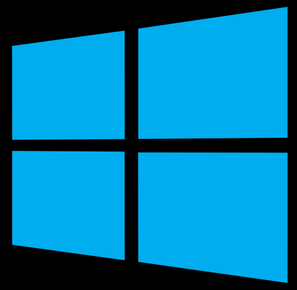
Coming Soon: Background Maintenance

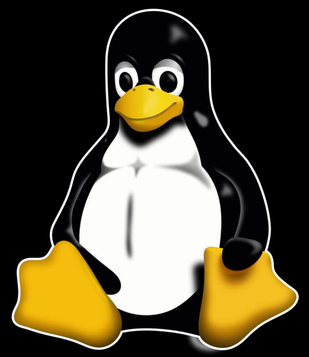


https://xkcd.com/303/


--filter=blob:none
git sparse-checkout init --cone
.gitignore
src
LFS

Scalar





https://github.com/microsoft/scalar

`scalar clone <url>` will...

- Create your repo in `<repo>/src`
- Default to using partial clone (`--filter=blob:none`)
- Default to using `sparse-checkout`
- Set up background maintenance
- Initialize recommended, advanced config settings

https://github.com/microsoft/scalar

Derrick Stolee

GitHub: @derrickstolee
Twitter: @stolee
Website: https://stolee.dev