

Version Control!

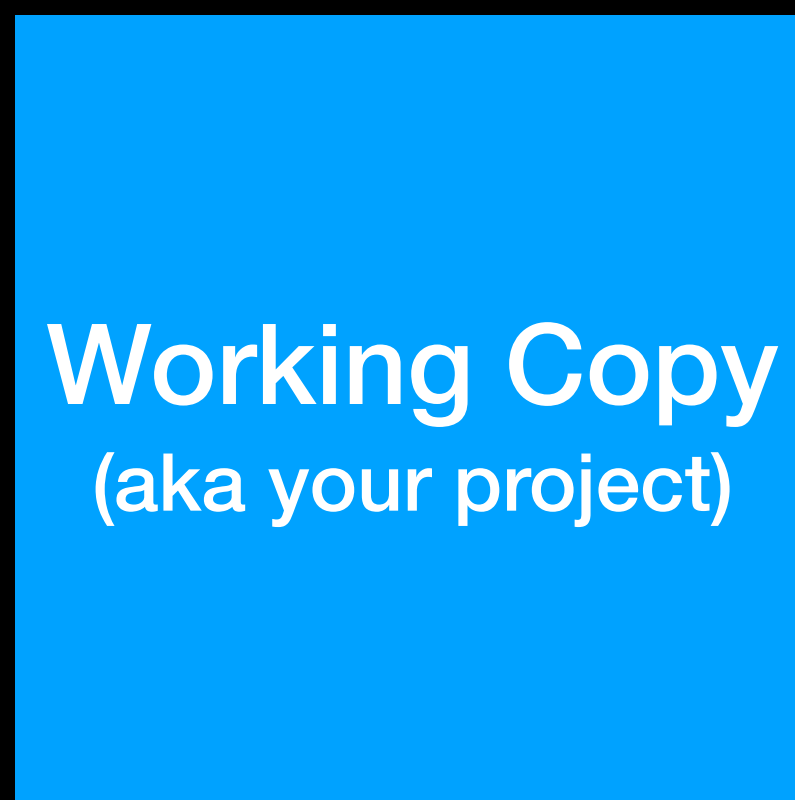
and what it's good for

What we'll cover today

1. Version control overview: what, why and how
2. Comparing technologies
3. git gud - core commands and concepts
4. Gotchas and footguns - general and Unity-specific
5. Practical workshop exercises - cloning a project, making changes and pushing them

What is version control, actually?

You do stuff here
(add, remove, edit files)



Working Copy
(aka your project)

And each time you
make changes you
want to save, you send
them here



Repository
(a saved history of
changes to your
project)



You can also fetch back
old versions of your work!

ok but y tho?

we all know how to use dropbox or google drive
already



CoolProject



CoolProject_v2



CoolProject_v3



CoolProject_v3_edit



CoolProject_v4_final



CoolProject_v4_final_fixed



CoolProject_v4_final_fixed2

game dev: a play in one act

YOU

I have a great idea to improve this **THING THAT IS DUE TOMORROW**, it should be pretty easy to add, I will change some code and edit some assets

THING THAT IS DUE TOMORROW

breaks horribly

YOU

oh no

Good things come in threes

1. Backups
2. Sharing/collaboration
3. Project history (i.e. “versions”)

STOP: vocab time!



“source control” and “version control” mean the same thing

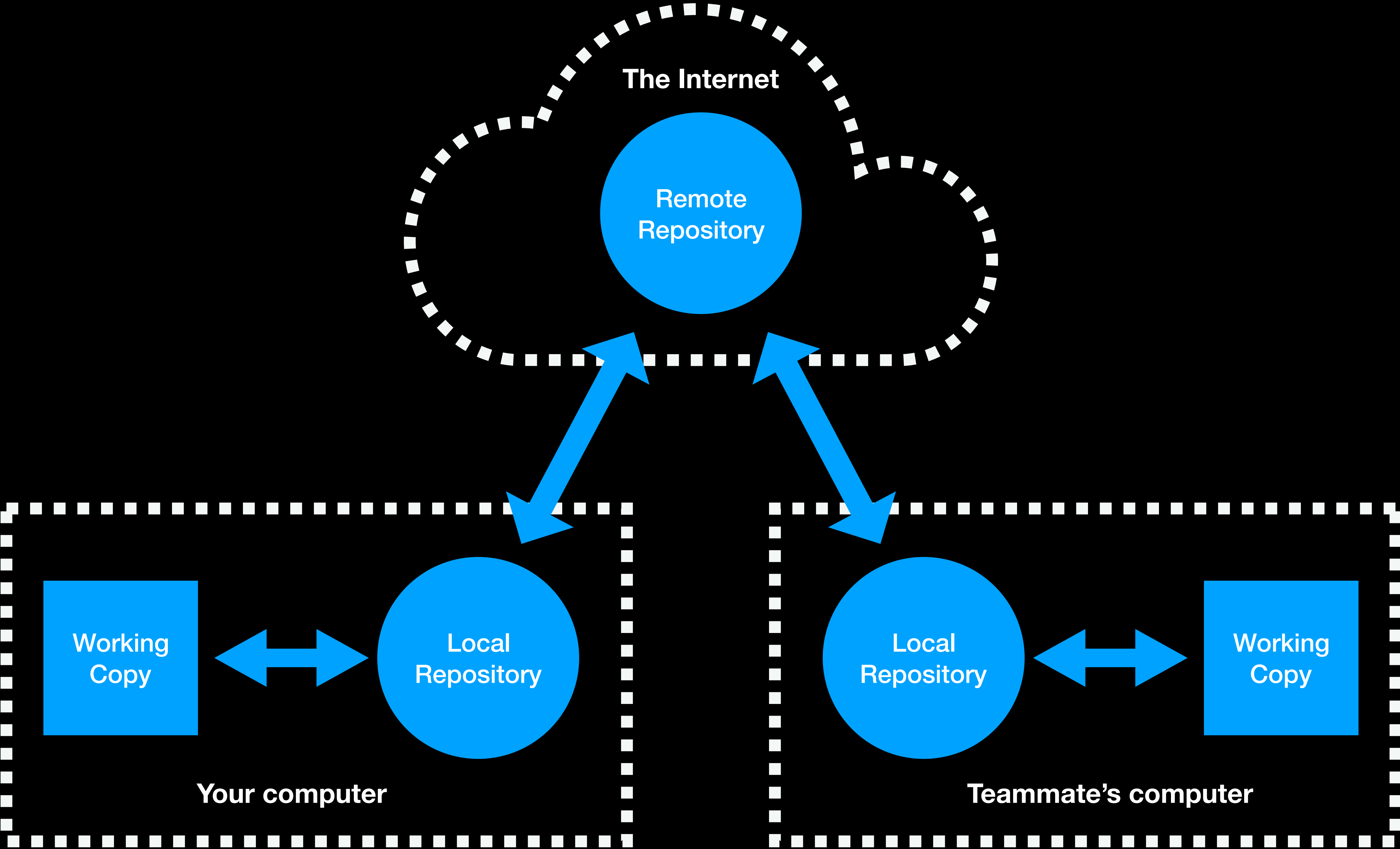
VCS stands for Version Control Software. These are all types of VCS:

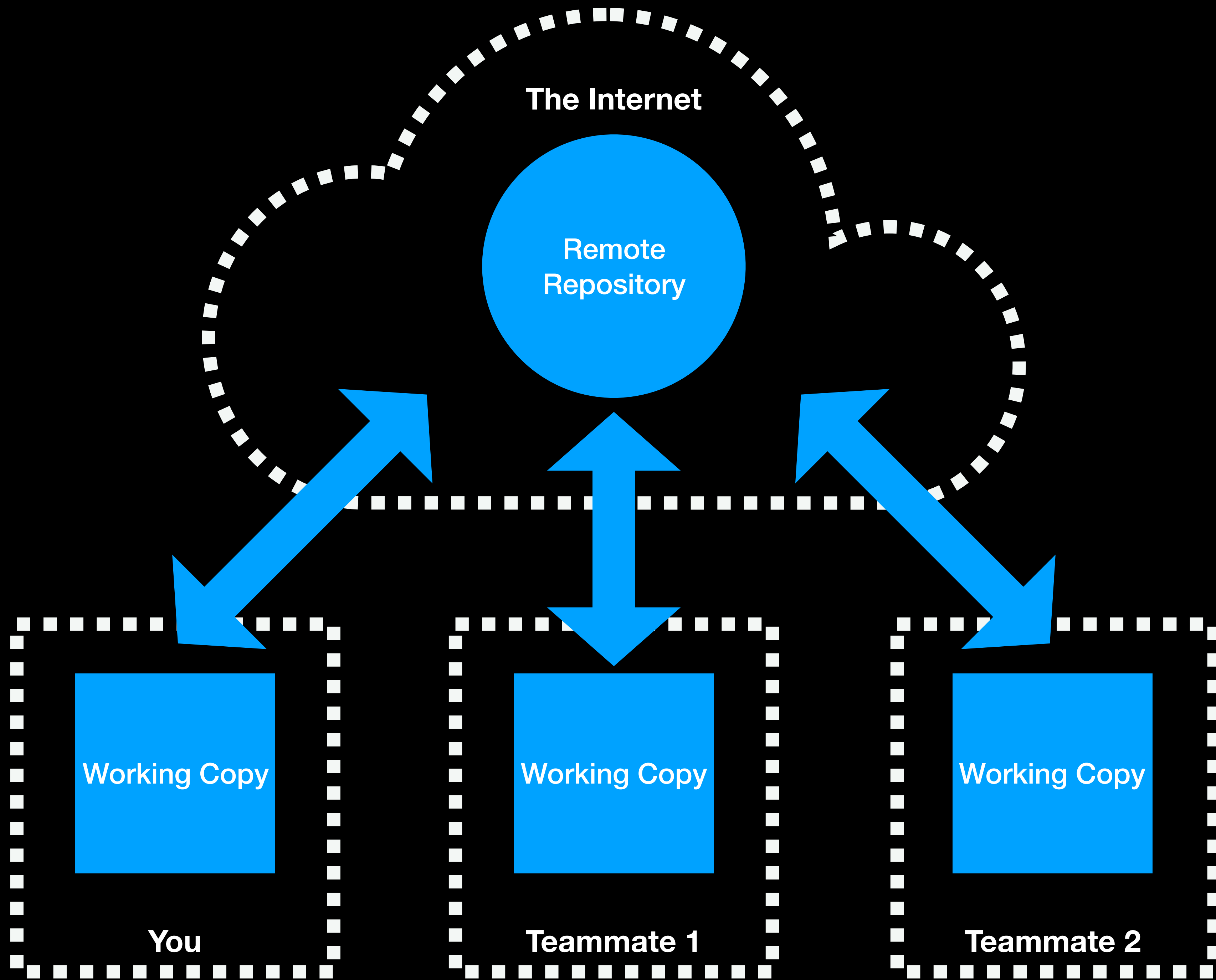
- git (not an acronym)
- svn (not an acronym; short for “Subversion”)
- Unity Collaborate (we’ll talk more about this in a moment)
- CVS (ok, this one is an acronym: Concurrent Versions System)
- Perforce, Plastic SCM (acronym!), Mercurial, ClearCase, etc, etc....

git vs Unity Collaborate

(ok but y tho, round 2)

	git (with free Github account)	Unity Collaborate (with Unity teams free account)
Works with non-Unity projects?		
Keeps project history?	Forever	90 days
Number of users	Unlimited for public repositories, 3 for private	3
Storage space	1GB recommended per repository, hard limit 100GB	1GB per team



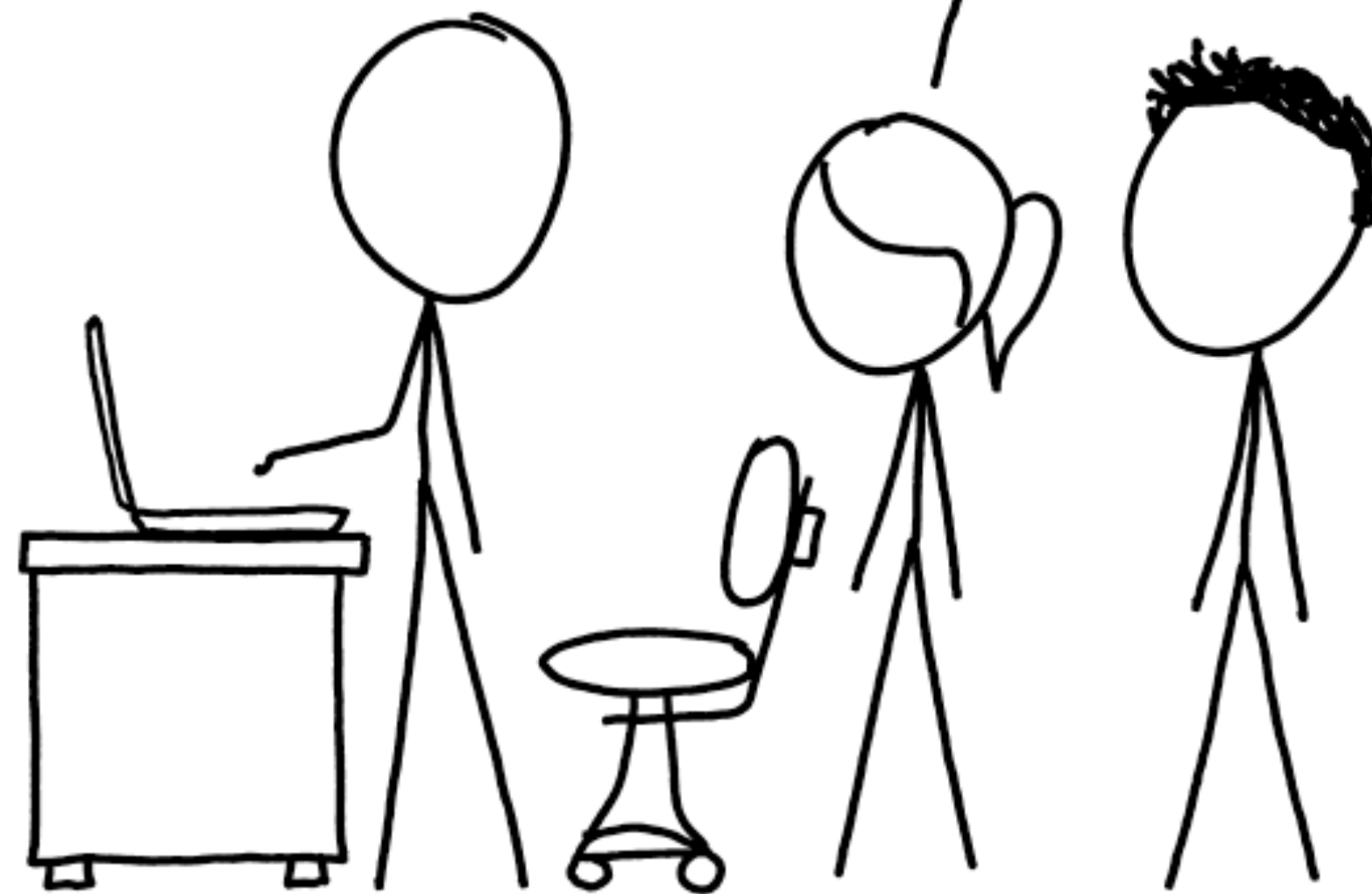


git gud (the basics)

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.



```
git status
```

```
git add -A
```

```
git commit -m "useful sensible commit message  
explaining what you've changed goes here"
```

```
git pull
```

```
git push
```




- Ready to push changes? commit -> pull (and merge) -> push
- Unity scenes and prefabs don't always merge well - you'll need some dogs licking windows or merge chickens
- Always add assets to a project via the Unity Editor
- Always check the project runs before committing
- TALK TO YOUR TEAM

**enough talking, let's
break a project**

**git better (handling
merge conflicts)**

Merge conflicts

```
git status
```

```
git checkout --ours <FILENAME>
```

or

```
git checkout --theirs <FILENAME>
```

then

```
git add -A
```

```
git commit -m "Fixed merge conflicts like a boss"
```

Too scary? Worried you'll break it?

```
git merge --abort
```

