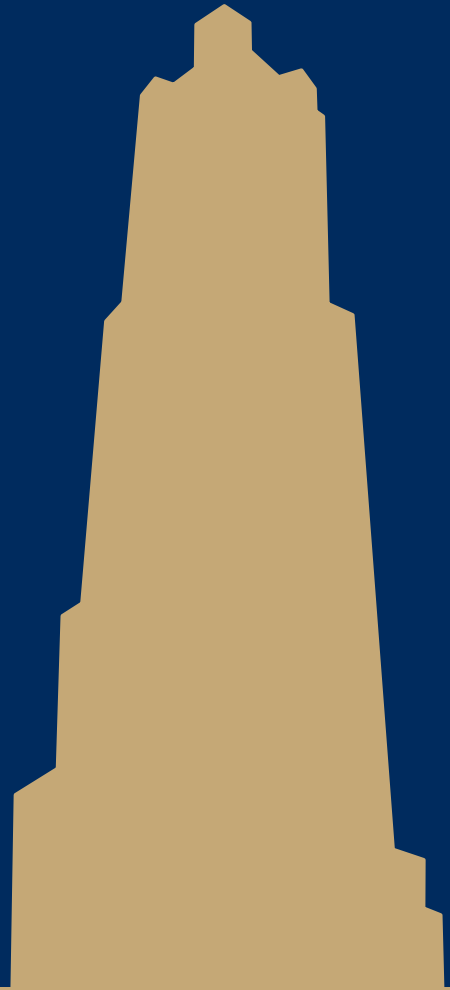


An introduction to version control systems with Git



Version control systems

Version control systems record changes to a file or set of files over time so that you can recall specific versions later

Many systems have risen to popularity over the years

RCS

CVS

Subversion

We will focus on Git

Why use version control?

These systems help with:

Tracking changes

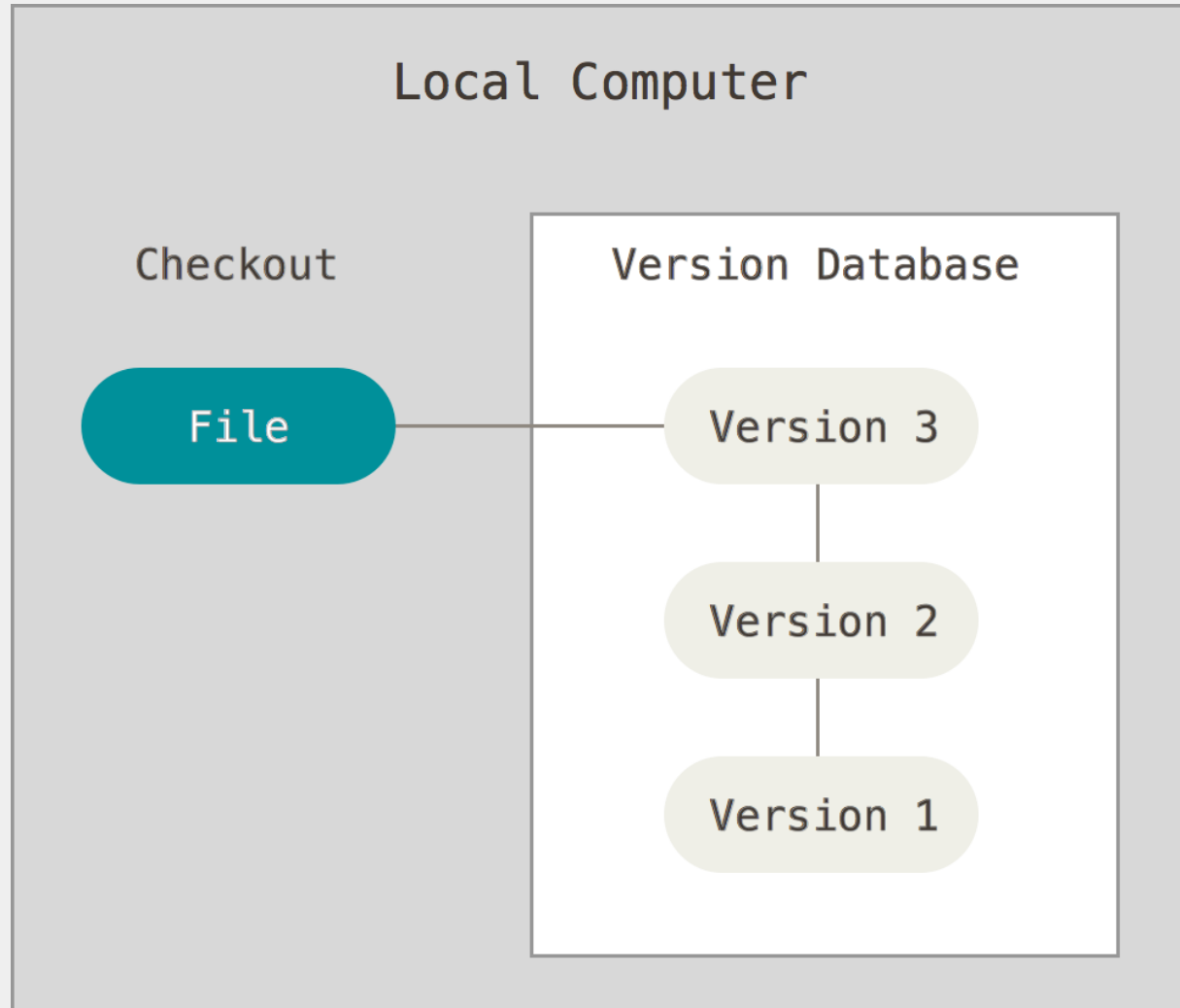
Short and long term undo

Backup and restore

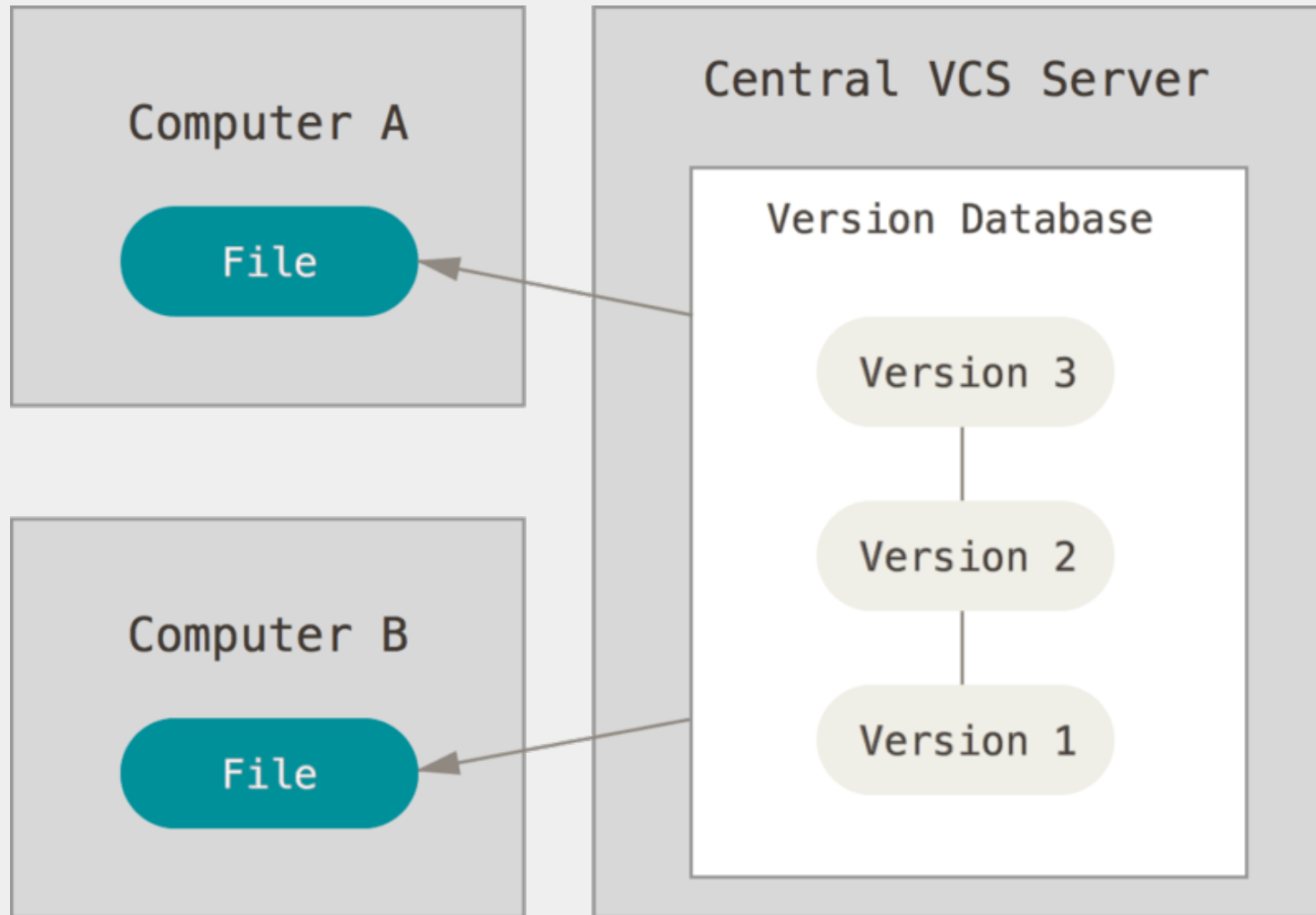
Synchronization

Collaboration

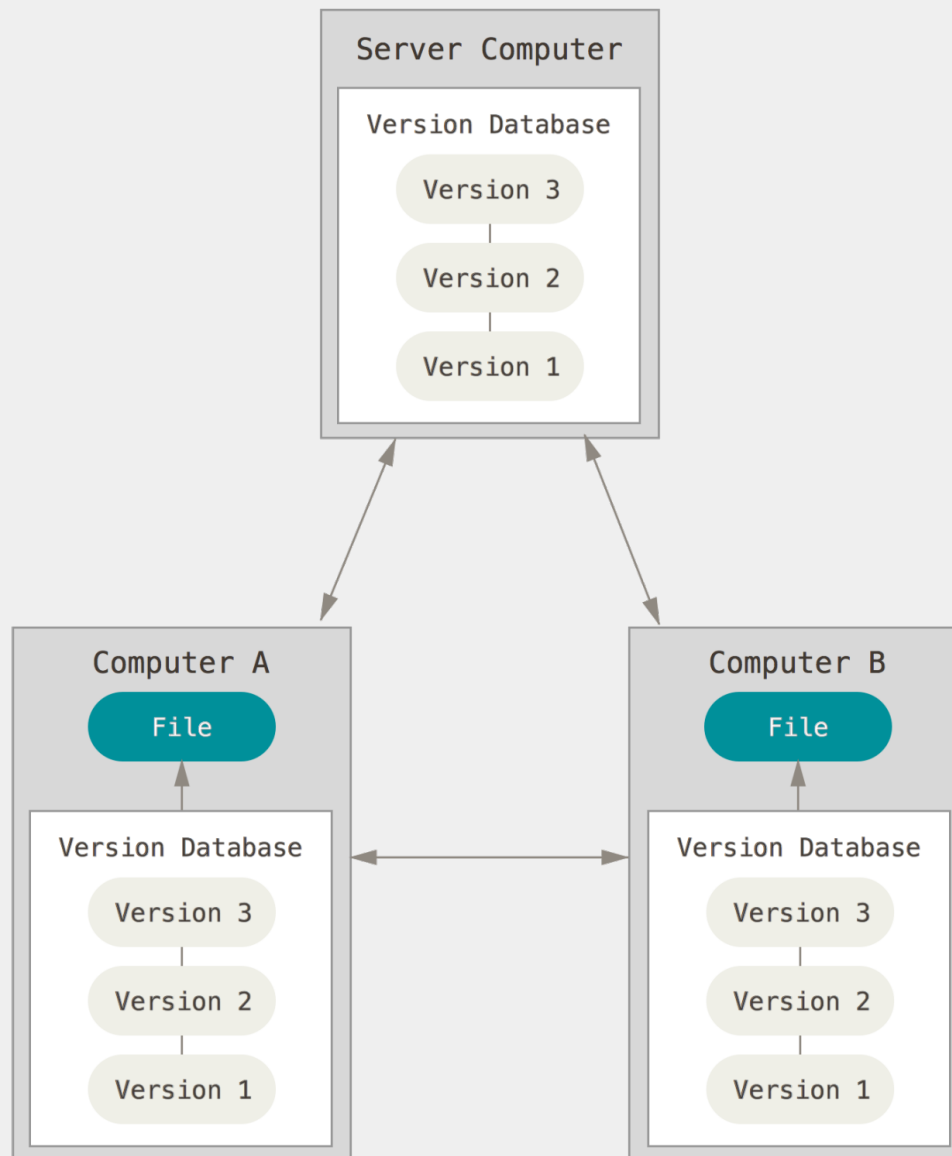
Local version control systems



Centralized version control systems



Distributed version control systems

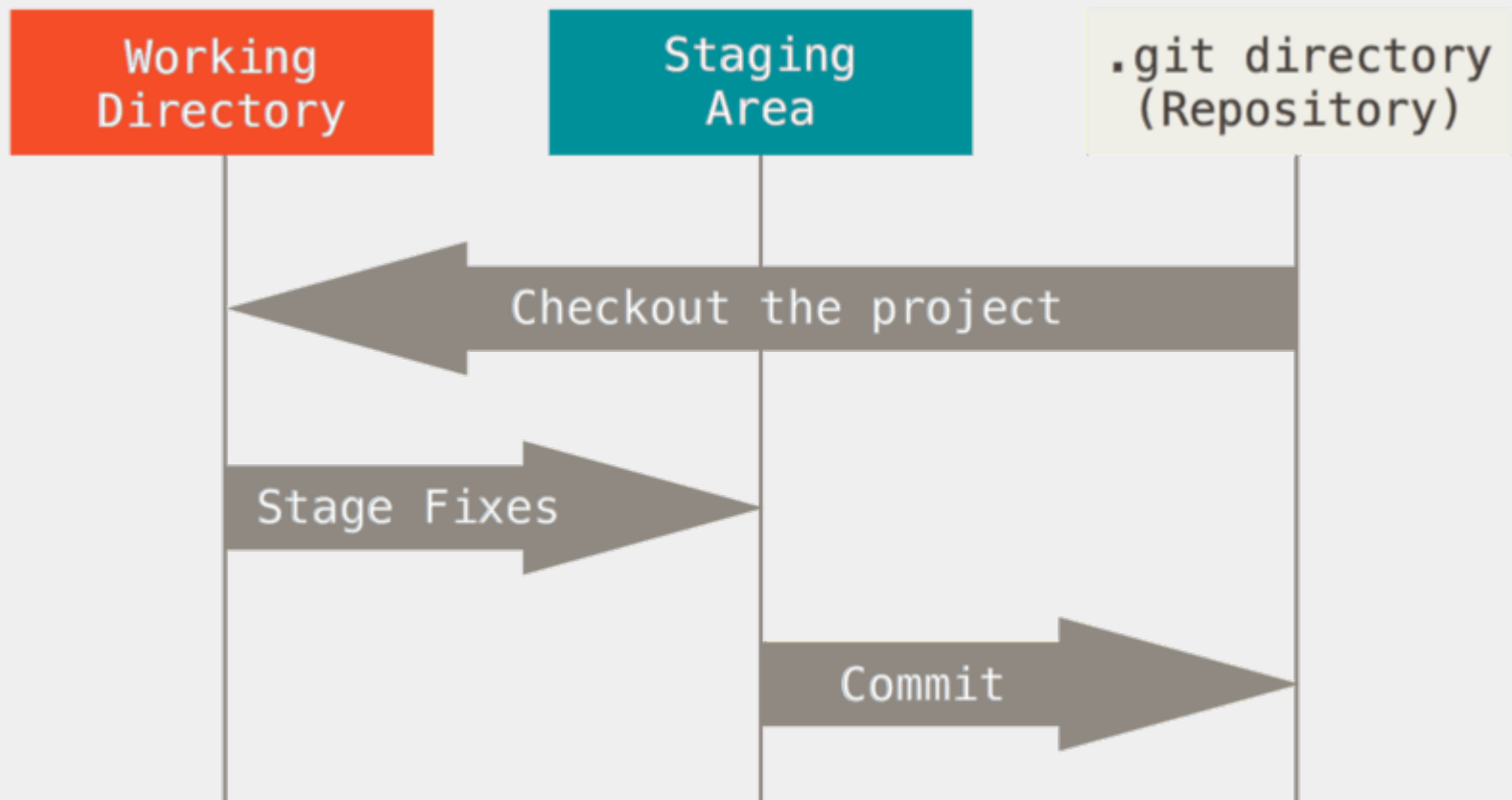


The basic Git workflow

Modify files in your *working directory*

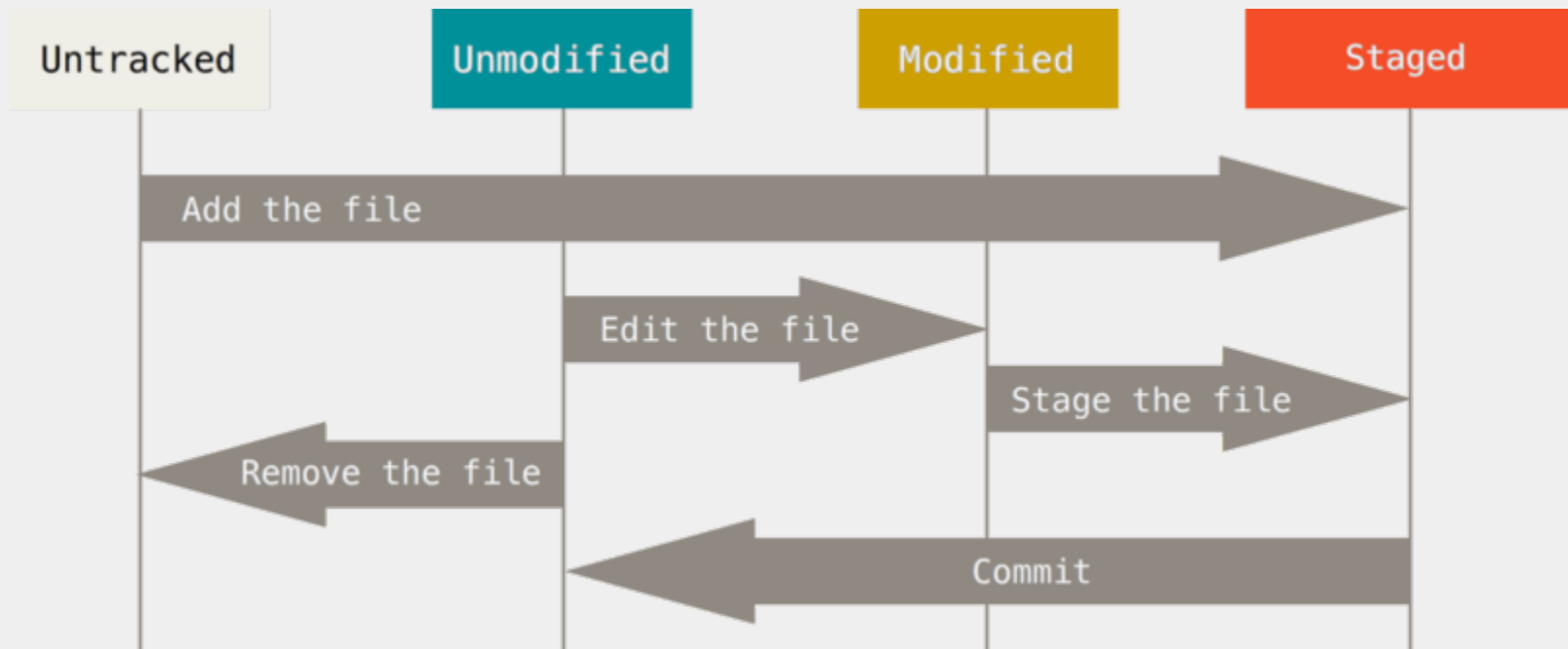
Stage the files, adding snapshots to your *staging area*

Commit your changes to your local copy of the *repository*

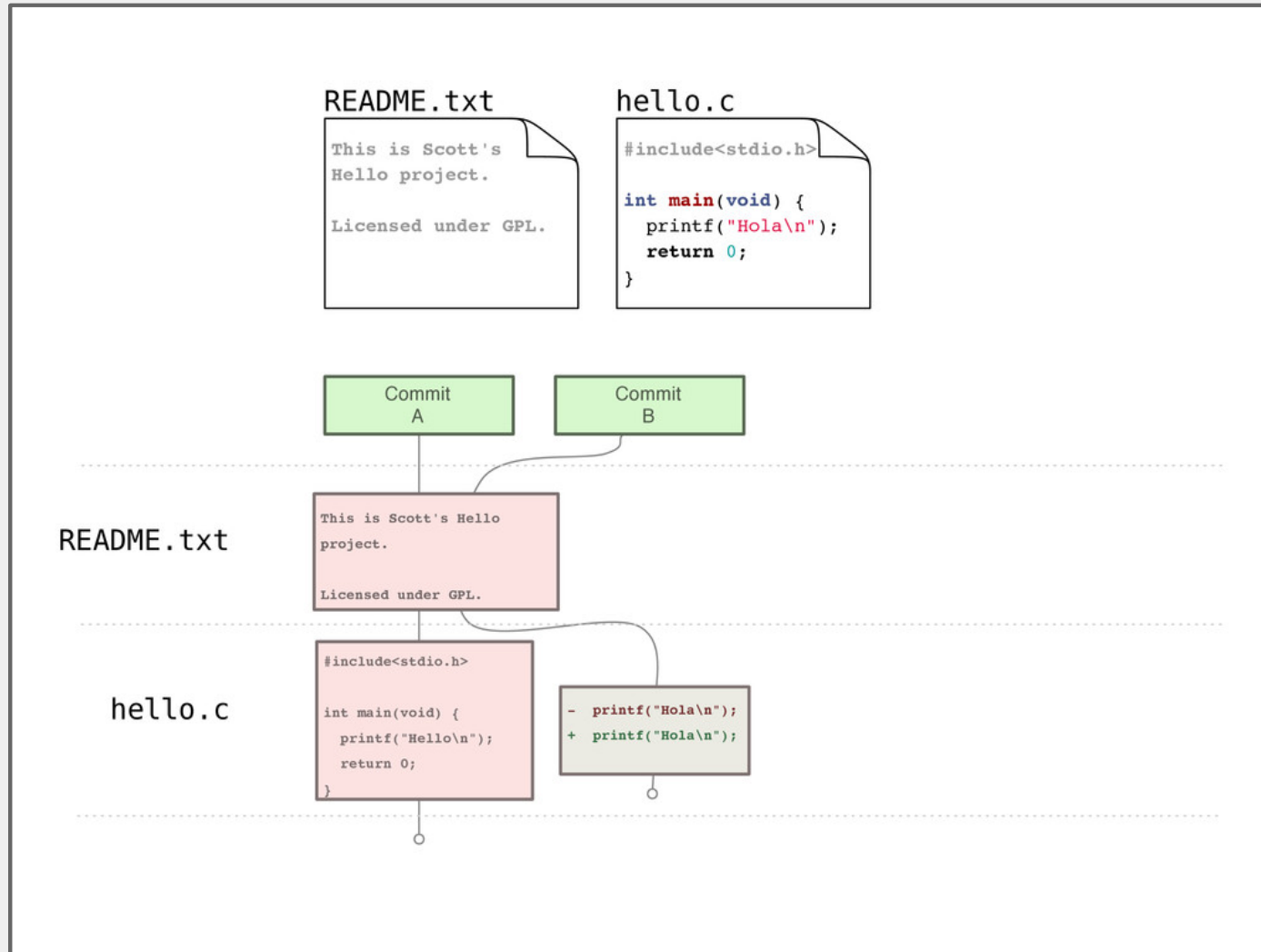


The lifecycle of a file in Git

Git does not necessary keep track of all files in your working directory



Example repository



Gitting started

Set your identity

```
$ git config --global user.name "John Doe"
```

```
$ git config --global user.email jdoe@example.com
```

Set other configuration options

```
$ git config --global color.ui true
```

Get help

```
$ git help <verb>
```

Creating a new repository

```
$ git init
```

Creates a new (empty) repository in the current directory

Copying a repository

For this class, your instructor will create a repository for you, you will just need to copy it from GitHub to your computer using the following command:

```
$ git clone <repository>
```

Creates a copy of `<repository>` in the current directory

Staging files

As you work, you will create new files and modify existing files,
when you are satisfied with your changes, you can stage them for
commit with:

```
$ git add <file_pattern>
```

Committing changes

Commits create a new version in the repository

Include a commit message describing the new version

```
$ git commit -m <msg>
```

Checking working directory status

```
$ git status
```

Reports:

Files in the working directory that are not tracked

File modifications not yet staged for commit

File additions and modifications staged for commit

Overviewing commit history

```
$ git log
```

Lists commits made to the current repository

Git example (cloning via GitHub)

Handy command - comparing versions

It may be handy to see exactly how files changed

```
$ git diff
```

Shows modifications not yet staged for commit

```
$ git diff <commit_id>
```

Show changes since the commit specified

```
$ git diff <commit_id1> <commit_id2>
```

Show changes between two commits

What we've covered here...

... presents only a brief overview of Git

Further topics:

branching

rebasing

tagging

...

Further resources:

<https://git-scm.com/book/en/v2>

<http://gitref.org/>

<http://gitimmersion.com/>