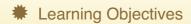


Version Control with Git

Creating a Repository



• Create a local Git repository.

Once Git is configured, we can start using it. Let's create a directory for our work and then move into that directory:

```
$ mkdir planets
$ cd planets
```

Then we tell Git to make planets a repository—a place where Git can store versions of our files:

```
$ git init
```

If we use ls to show the directory's contents, it appears that nothing has changed:

```
$ ls
```

But if we add the —a flag to show everything, we can see that Git has created a hidden directory within planets called .git:

```
$ ls -a
```

```
. .. .git
```

Git stores information about the project in this special sub-directory. If we ever delete it, we will lose the project's history.

We can check that everything is set up correctly by asking Git to tell us the status of our project:

```
$ git status
```

```
# On branch master
# Initial commit
nothing to commit (create/copy files and use "git add" to track)
```

Places to Create Git Repositories

Dracula starts a new project, moons, related to his planets project. Despite Wolfman's concerns, he enters the following sequence of commands to create one Git repository inside another:

```
cd
             # return to home directory
mkdir planets # make a new directory planets
cd planets  # go into planets
git init  # make the planets directory a Git repository
git init
           # make the moons sub-directory a Git repository
```

Why is it a bad idea to do this? How can Dracula "undo" his last git init?

Software Carpentry | Source | Contact | License