

Projects under version control

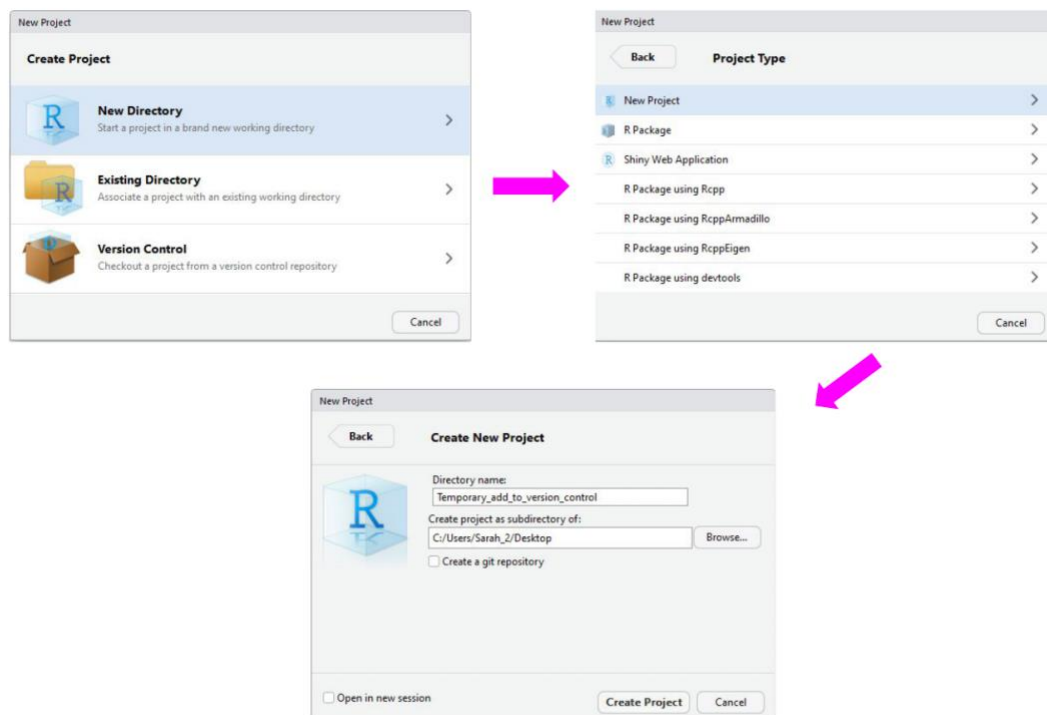
In the previous lesson, we linked RStudio with Git and GitHub. In doing this, we created a repository on GitHub and linked it to RStudio. Sometimes, however, you may already have an R Project that isn't yet under version control or linked with GitHub. Let's fix that!

Linking an existing Project with Git

So what if you already have an R Project that you've been working on, but don't have it linked up to any version control software (tut tut!)?

Thankfully, RStudio and GitHub recognize this can happen and have steps in place to help you (admittedly, this is slightly more troublesome to do than just creating a repository on GitHub and linking it with RStudio before starting the project...).

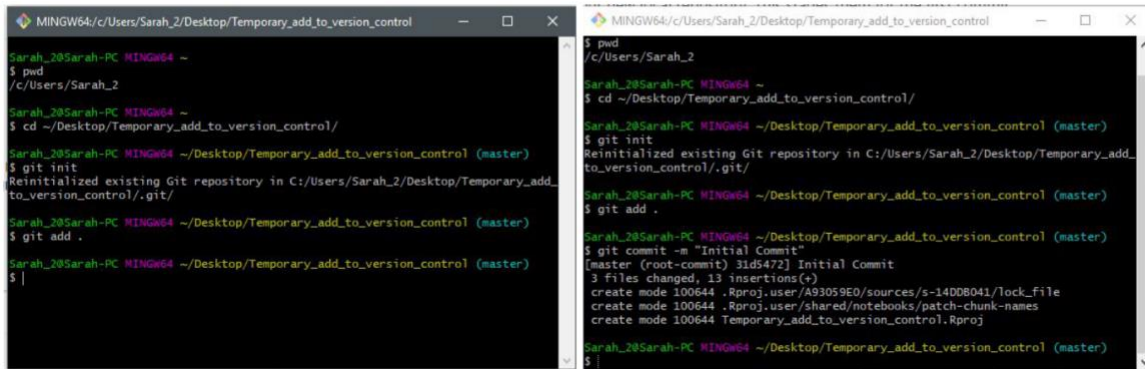
So first, let's set up a situation where we have a local project that isn't under version control. Go to File > New Project > New Directory > New Project and name your project. Since we are trying to emulate a time where you have a project not currently under version control, do **NOT** click "Create a git repository". Click Create Project.



Creating a project that is not under version control

We've now created an R Project that is not currently under version control. Let's fix that. First, let's set it up to interact with Git. Open Git Bash or Terminal and navigate to the directory containing your project files. Move around directories by typing `cd ~/dir/name/of/path/to/file`

When the command prompt in the line before the dollar sign says the correct directory location of your project, you are in the correct location. Once here, type `git init` followed by `git add .` - this initializes (*init*) this directory as a git repository and *adds* all of the files in the directory (.) to your local repository. Commit these changes to the git repository using `git commit -m "Initial commit"`



```
MINGW64/c/Users/Sarah_2/Desktop/Temporary_add_to_version_control
Sarah_2@Sarah-PC MINGW64 ~
$ pwd
/c/Users/Sarah_2
Sarah_2@Sarah-PC MINGW64 ~
$ cd ~/Desktop/Temporary_add_to_version_control/
Sarah_2@Sarah-PC MINGW64 ~/Desktop/Temporary_add_to_version_control (master)
$ git init
Reinitialized existing Git repository in C:/Users/Sarah_2/Desktop/Temporary_add_to_version_control/.git/
Sarah_2@Sarah-PC MINGW64 ~/Desktop/Temporary_add_to_version_control (master)
$ git add .
Sarah_2@Sarah-PC MINGW64 ~/Desktop/Temporary_add_to_version_control (master)
$ |

MINGW64/c/Users/Sarah_2/Desktop/Temporary_add_to_version_control
$ pwd
/c/Users/Sarah_2
Sarah_2@Sarah-PC MINGW64 ~
$ cd ~/Desktop/Temporary_add_to_version_control/
Sarah_2@Sarah-PC MINGW64 ~/Desktop/Temporary_add_to_version_control (master)
$ git init
Reinitialized existing Git repository in C:/Users/Sarah_2/Desktop/Temporary_add_to_version_control/.git/
Sarah_2@Sarah-PC MINGW64 ~/Desktop/Temporary_add_to_version_control (master)
$ git add .
Sarah_2@Sarah-PC MINGW64 ~/Desktop/Temporary_add_to_version_control (master)
$ git commit -m "Initial Commit"
[master (root-commit) 31d5472] Initial Commit
3 files changed, 13 insertions(+)
create mode 100644 .Rproj.user/A93059E0/sources/s-140D8041/lock_file
create mode 100644 .Rproj.user/shared/notebooks/patch-chunk-names
create mode 100644 Temporary_add_to_version_control.Rproj
Sarah_2@Sarah-PC MINGW64 ~/Desktop/Temporary_add_to_version_control (master)
$ |
```

Linking the project folder with Git so it is now under version control

At this point, we have created an R Project and have now linked it to Git version control. The next step is to link this with GitHub.

Linking this project with GitHub

To do this, go to GitHub.com, and again, create a new repository:

- 1) Make sure the name is the **exact same** as your R project;
- 2) Do **NOT** initialize a README file, .gitignore, or license.

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: JaneEverydayDoe / Repository name: Temporary_add_to_version_control ✓

Great repository names are short and memorable. Need inspiration? How about [bug-free-disco](#).

Description (optional): Linking a pre-existing R Project with GitHub

☒ **Public**
Anyone can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

☒ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None Add a license: None ⓘ

Create repository

Creating a repository on GitHub that is named the same as your R project

Upon creating the repository, you should see a page like this:

JaneEverydayDoe / Temporary_add_to_version_control

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Quick setup — if you've done this kind of thing before

Set up in Desktop or **HTTPS** `git@github.com:JaneEverydayDoe/Temporary_add_to_version_control.git`

We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line

```
echo "# Temporary_add_to_version_control" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin git@github.com:JaneEverydayDoe/Temporary_add_to_version_control.git
git push -u origin master
```

...or push an existing repository from the command line

```
git remote add origin git@github.com:JaneEverydayDoe/Temporary_add_to_version_control.git
git push -u origin master
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

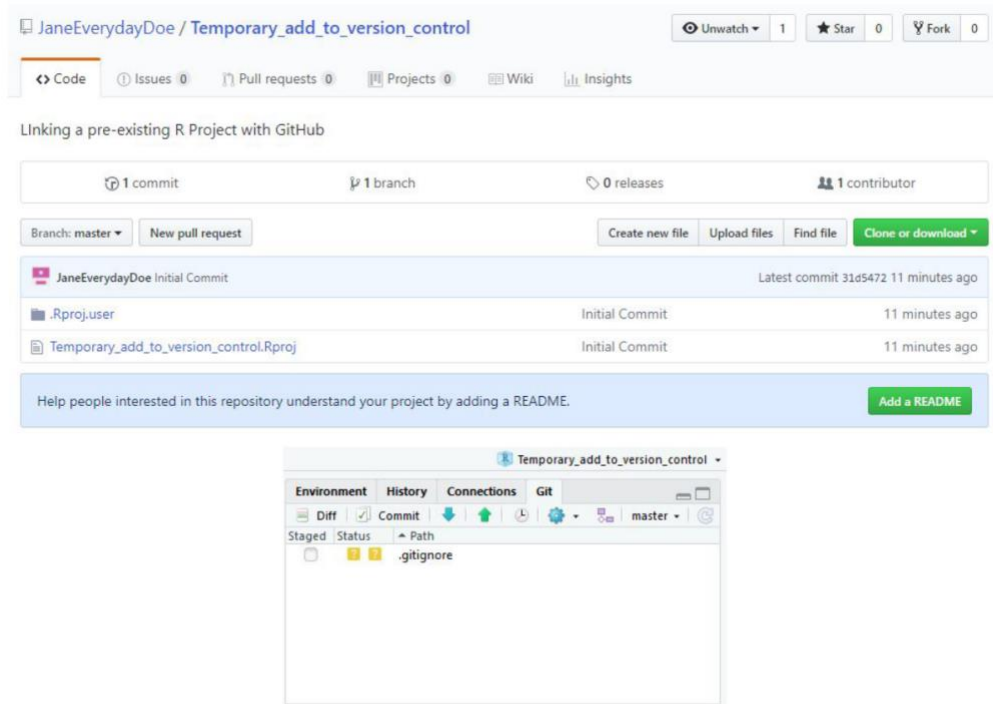
[Import code](#)

ProTip! Use the URL for this page when adding GitHub as a remote.

Your new repository on GitHub containing code to push from the command line

You should see that there is an option to “Push an existing repository from the command line” with instructions below containing code on how to do so. In Git Bash or Terminal, copy and paste these lines of code to link your repository with GitHub. After doing so, refresh your GitHub page and it should now look something like the image below.

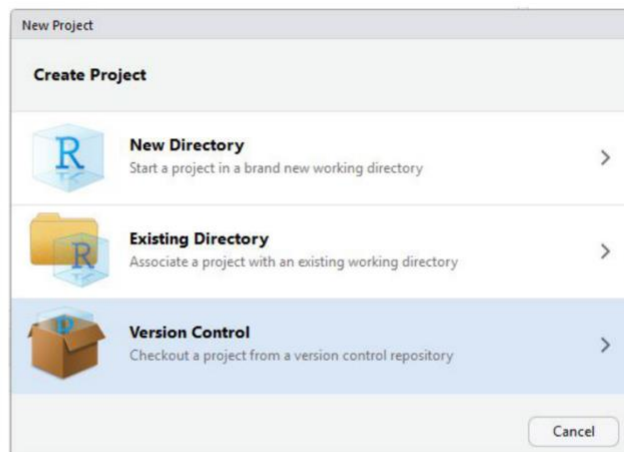
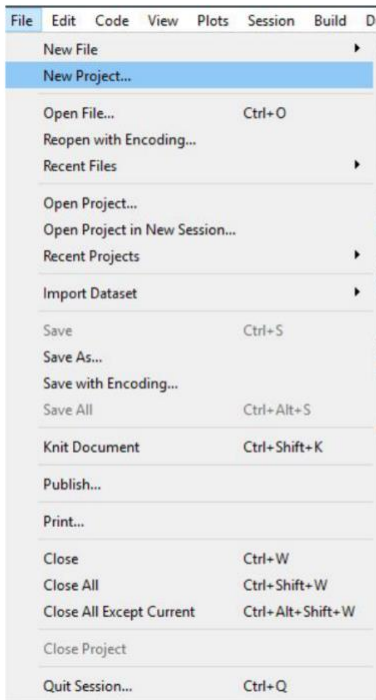
When you re-open your project in RStudio, you should now have access to the Git tab in the upper right quadrant and can push to GitHub from within RStudio any future changes.



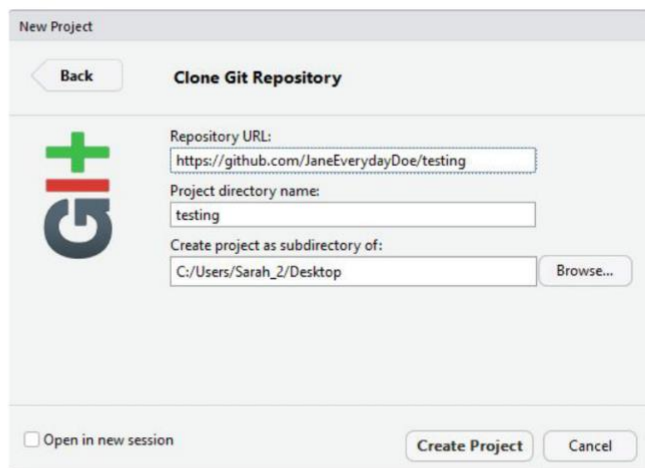
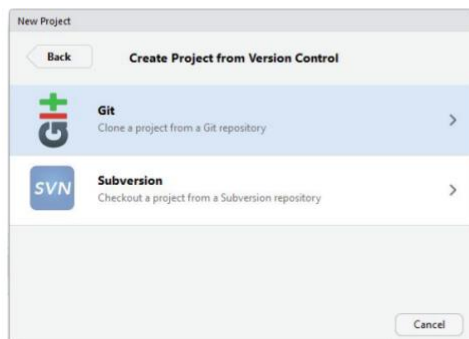
You've now pushed your R project repository to your GitHub repository of the same name

Working on an existing GitHub repository

If there is an existing project that others are working on that you are asked to contribute to, you can link the existing project with your RStudio. It follows the exact same premise as that from the last lesson where you created a GitHub repository and then cloned it to your local computer using RStudio. In brief, in RStudio, go to File > New Project > Version Control. Select Git as your version control system, and like in the last lesson, provide the URL to the repository that you are attempting to clone and select a location on your computer to store the files locally. Create the project.



Follow the same steps as previously done to clone your own repository to a new project in RStudio



Clone an existing project from GitHub from within RStudio

All the existing files in the repository should now be stored locally on your computer and you have the ability to push edits from your RStudio interface. The only difference from the last lesson is that you did not create the original repository, instead you cloned somebody else's.

Summary

In this lesson, we went over how to convert an existing project to be under Git version control using the command line. Following this, we linked your newly version controlled project to GitHub using a mix of GitHub commands and the command line. We then briefly recapped how to clone an existing GitHub repository to your local machine using RStudio.