How to use Git with R and RStudio

Fri, Aug 19, 2016 reproducible research, git, github, r, rstudio

This tutorial in the context of the **Reproducible Research Workshop** provides you with the first steps on how to use Git with R and RStudio. (The tutorial was originally created on GitHub and hosted here.)

Objectives of this tutorial:

- Set up and install Git
- Set up Git in RStudio
- Create new Git project in RStudio
- Clone/fork an existing project from GitHub
- Make some commits to your own project.

Motivation

R in combination with the distributed version control system *Git* provides a convenient setup to make your research project reproducible. Git allows you to track and share your code and analysis.

Some reasons to use version control are:

- It makes sharing of your projects *easy* (once it's setup, you'll get there)
- It facilitates collaboration. People can contribute to your project and vice-versa. You can also report errors (bugs) or suggest new additions (features) to projects.
- You can revert back to a previous version, if you find errors or accidently deleted something.
- You can *see* what changes between different versions of your code, analysis or written text!
- In R it makes sharing of your packages easy. And you can install development packages of others with two lines of code. `install.packages("devtools"); devtools::install_github("username/packagename")` (Development of R packages is more advanced in R, but is a well-structured way to keep your projects tidy; see: R Packages by Hadley Wickham)

GitHub is a user-friendly webservice that allows you to store your project repository remotely. Alternatives are gitlab and bitbucket.

RStudio integrates support for git, hence we are going to use the widely used combination *R + Git + RStudio*.

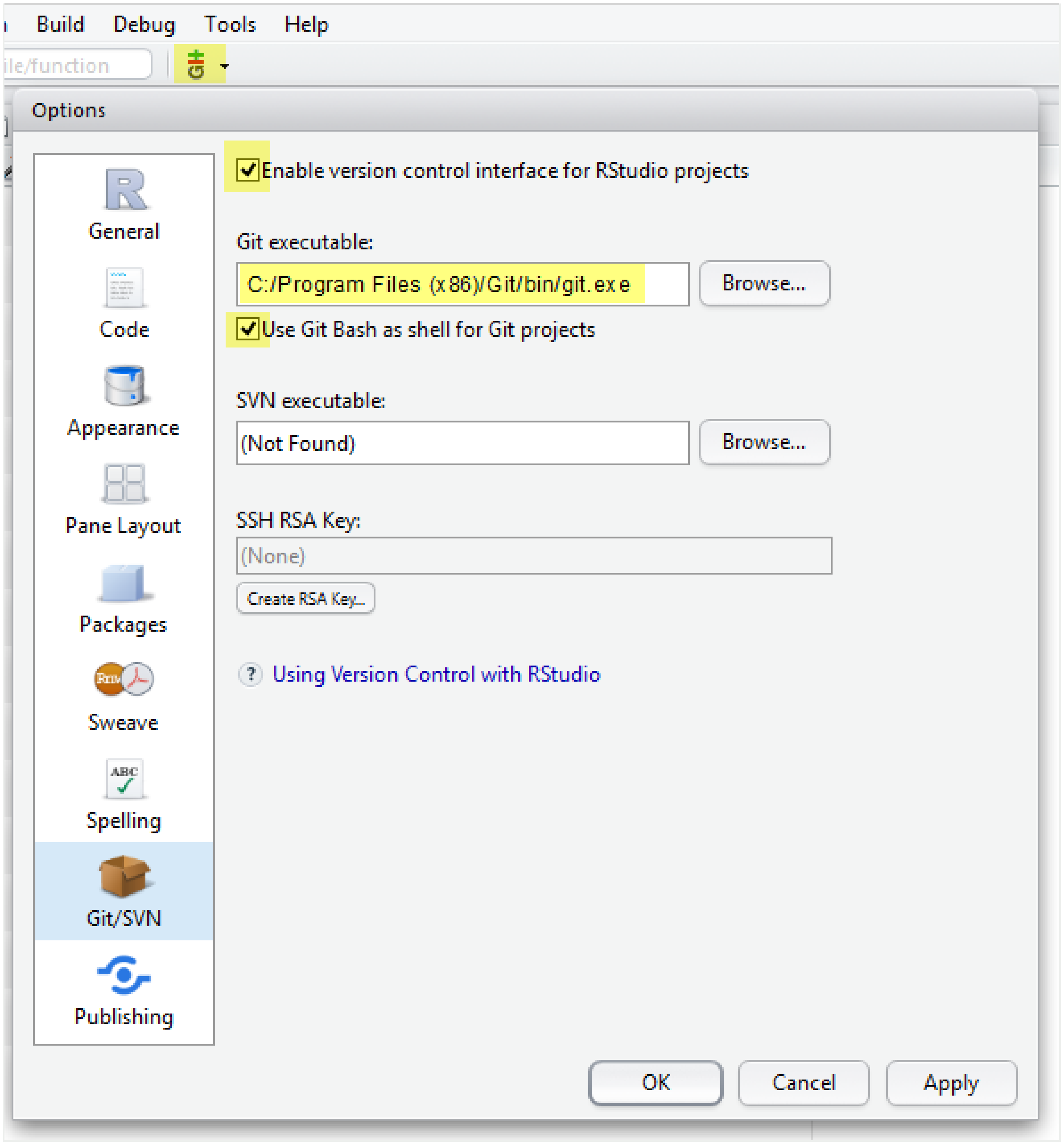
Part 1: Installation and setup

1. Installation: To get started you need the following software installed on your computer: Git and if you are new to R, then you also need to install R and RStudio. Additionally you will also need a GitHub account.

1. **Git (Download Git):** Download and install Git, making a note of where on your computer you are install it. *Optional Git clients: SourceTree or GitHub Desktop*.
2. **R (Download R):** Download and Install R (if not already installed).
3. **RStudio (Download RStudio Desktop):** Download and Install RStudio (if not already installed)
4. **GitHub account**: On GitHub create yourself a free GitHub account. *If you are new to Git follow the 15 min TryGit Tutorialto get a quick introduction to Git.*

2. Setup Git in RStudio: Tell RStudio where to find the Git installation.

1. Open RStudio and go to *Tools* > *Global Options*… click on *Git/SVN*
2. **Check** *Enable version control interface for RStudio projects*
3. **Set the path to the Git executable** that you just installed. Open a shell, if you don’t know where Git is installed.
   Windows: type `where git` and hit enter. The path should be something like: `C:/Program Files (x86)/Git/bin/git.exe`
   Linux/OS X: type `which git` and hit enter. The path should be something like: `/usr/bin/git`
4. **Restart RStudio**, if it worked out you will find the Git icon on the top toolbar, as shown below.



[✓] Enable version control interface for RStudio projects

Git executable:
C:/Program Files (x86)/Git/bin/git.exe

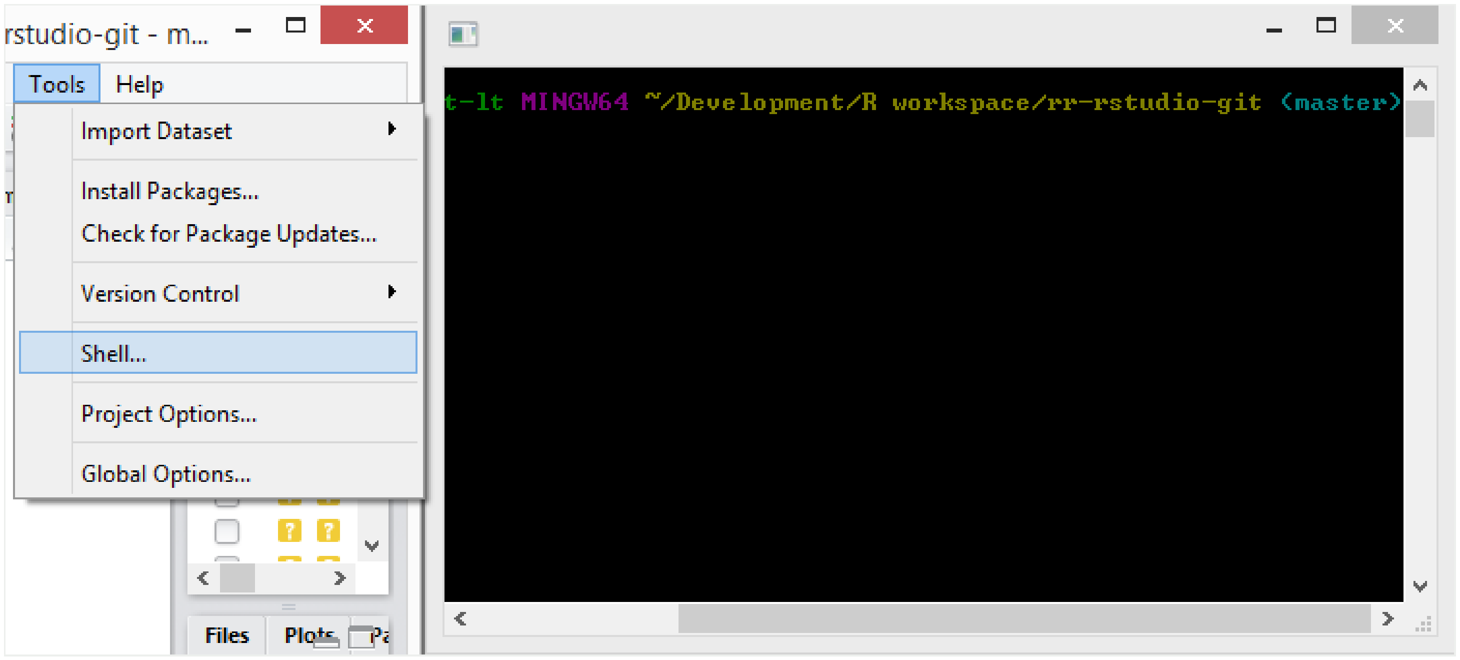
[✓] Use Git Bash as shell for Git projects

SVN executable:
(Not Found)

SSH RSA Key:
(None)

3. Setup Git: Configure Git and set your *user name* and *email* (the email address you used to register on GitHub). You can directly open the Git prompt from within RStudio. User name and email needs to be set only once. Go to *Tools* > *Shell* to open the Git Shell to tell Git your username and GitHub email.

```
git config --global user.name 'yourGitHubUsername'
git config --global user.email 'name@provider.com'
```



```
t-lt MINGW64 ~/Development/R workspace/rr-rstudio-git (master)
```

Part 2: Create a new RStudio project with Git

There are three ways to create version control for a RStudio project.

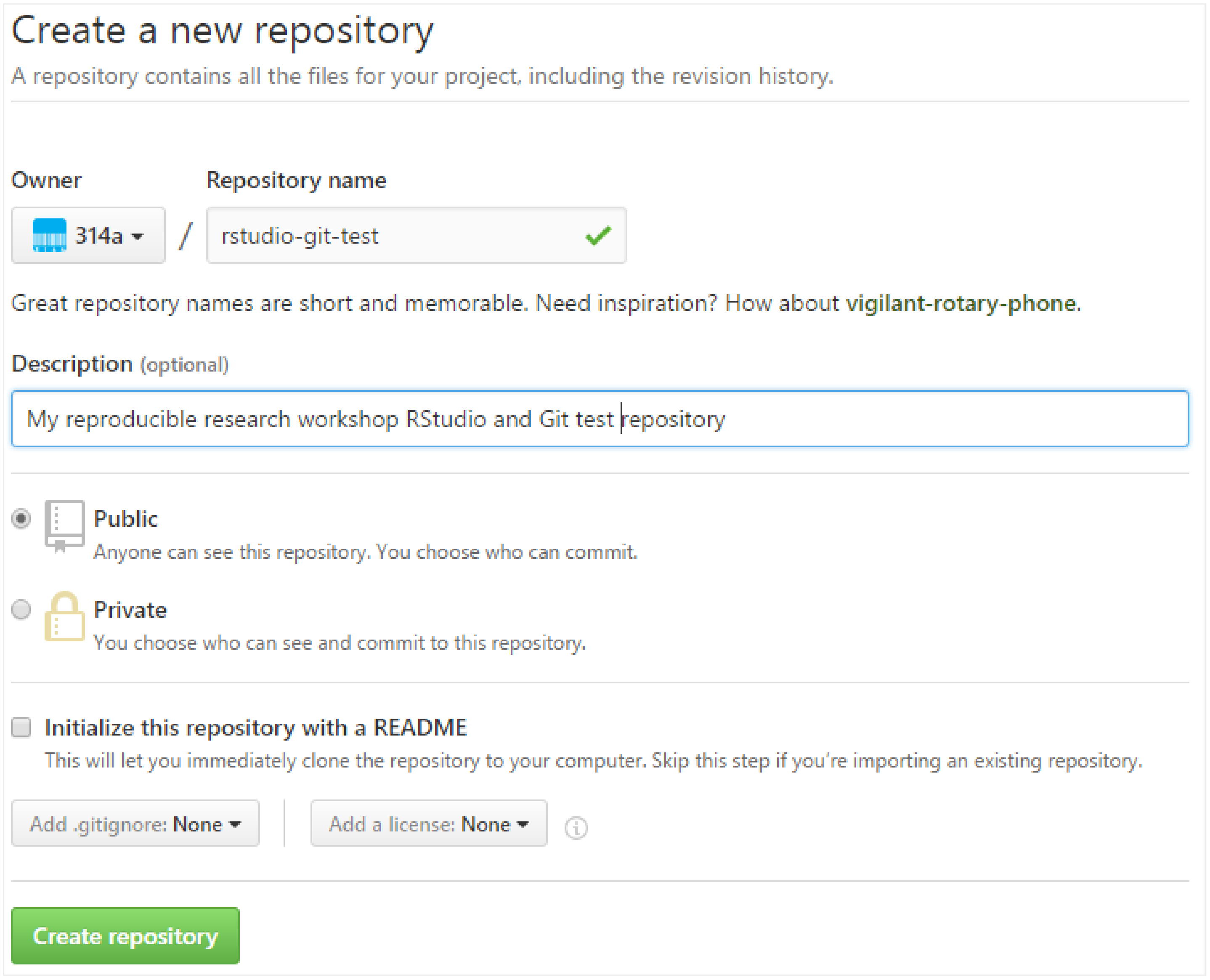
a) **Create a new project and create a local Git repository:** Select *File* > *New Project..*, create a project from a *New Directory* and check the option *Create a git repository*. In order to push to a remote repository later on you add that remote repository by using the Git shell. If you already know which online repository you want to use for your projects, option c) is more convenient.

b) **Create a new project from a folder under version control:** In this case you only need to create a new RStudio project for that directory and version control is automatically enabled. Go to *File > New Project*, select create a new project from an *Existing Directory*and create the project.

c) **Create a new project based on a remote Git repository:** Select *File > New Project..* and from the opening menu select to create a new project from *Version Control*, Choose Git, then provide the repository url (use the https link of the url if you want to avoid all the ssh trouble) from the the repository you want to clone and create the project.

In this tutorial we **create a project based on a remote GitHub repository** (option c). Hence we first create a new repository on GitHub and create our GitHub project from that repository.

1. Create a new GitHub repository: Login to your GitHub account and create a new GitHub repository. Give your new repository a short and memorable name e.g. `rstudio-git-test` , check the option to initialize this repository with a README and create the repository.


Create a new repository

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

Owner: 314a / Repository name: rstudio-git-test ✓

Great repository names are short and memorable. Need inspiration? How about **vigilant-rotary-phone**.

Description (optional)

My reproducible research workshop RStudio and Git test repository

[✓ Public] Anyone can see this repository. You choose who can commit.

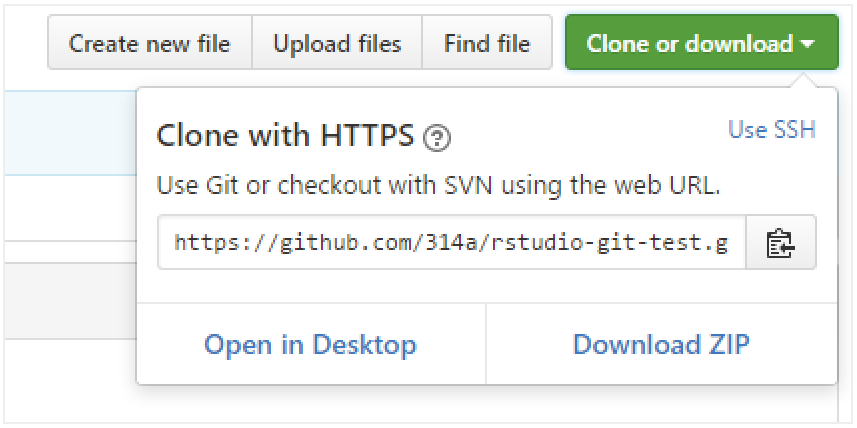
Private: You choose who can see and commit to this repository.

☐ 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

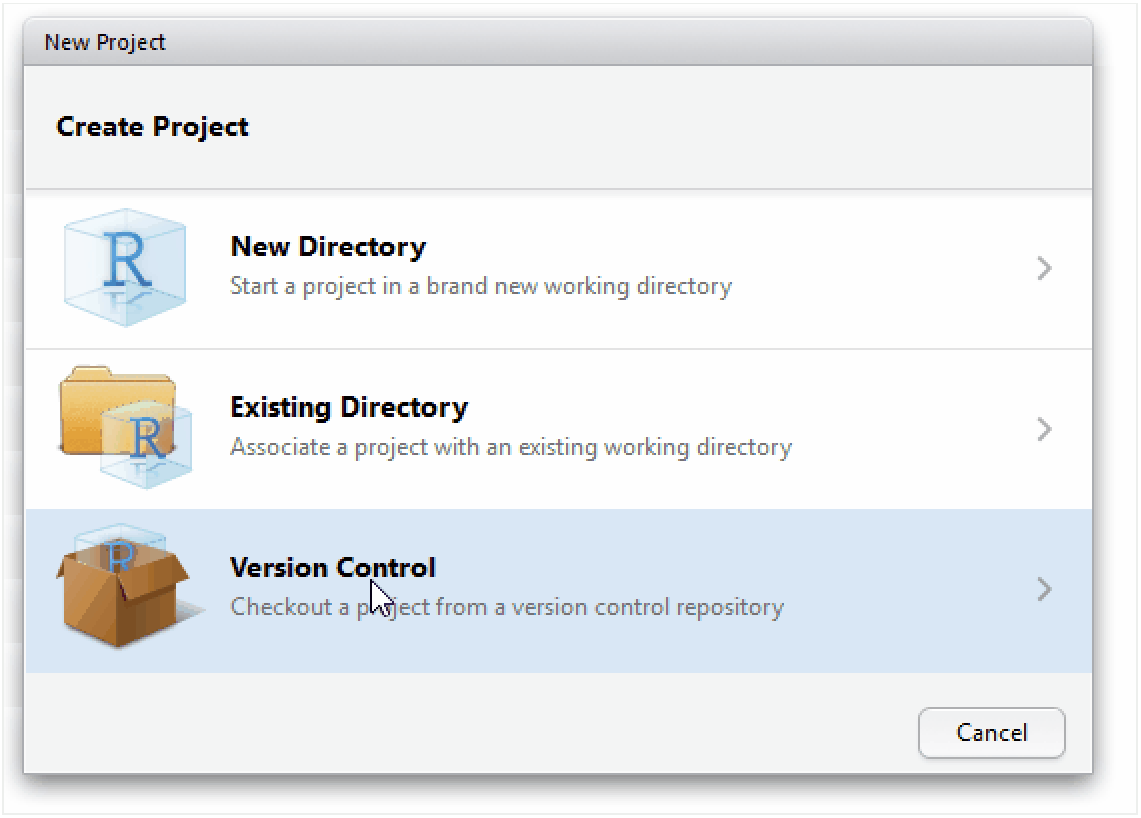
2. Copy the repository HTTPS url: To create a new Git based project in RStudio, we need the repository url. You find the repository HTTPS url on the just created GitHub project page. There press the button *Clone or download* and copy the HTTPS link of the project by clicking the little icon to the right of the URL. The link will be something like `https://github.com/yourusername/rstudio-git-test.git` .



Use Git or checkout with SVN using the web URL.

```
https://github.com/314a/rstudio-git-test.g
```

3. Create a new RStudio project with Git version control: Now everything is ready to create a new project with Git version control in RStudio. In RStudio Select *File* > *New Project..*, select *Version Control*, Choose *Git*, then provide the repository HTTPS link, select the R workspace folder and create the project. RStudio now copies (*clone* in Git terms) the content of the repository to your project folder. The content of the GitHub repository should now appear in the Files pane of RStudio and you should see there the created `README.md` .


New Project
Create Project
New Directory
Start a project in a brand new working directory
Existing Directory
Associate a project with an existing working directory
Version Control
Checkout a project from a version control repository
Cancel

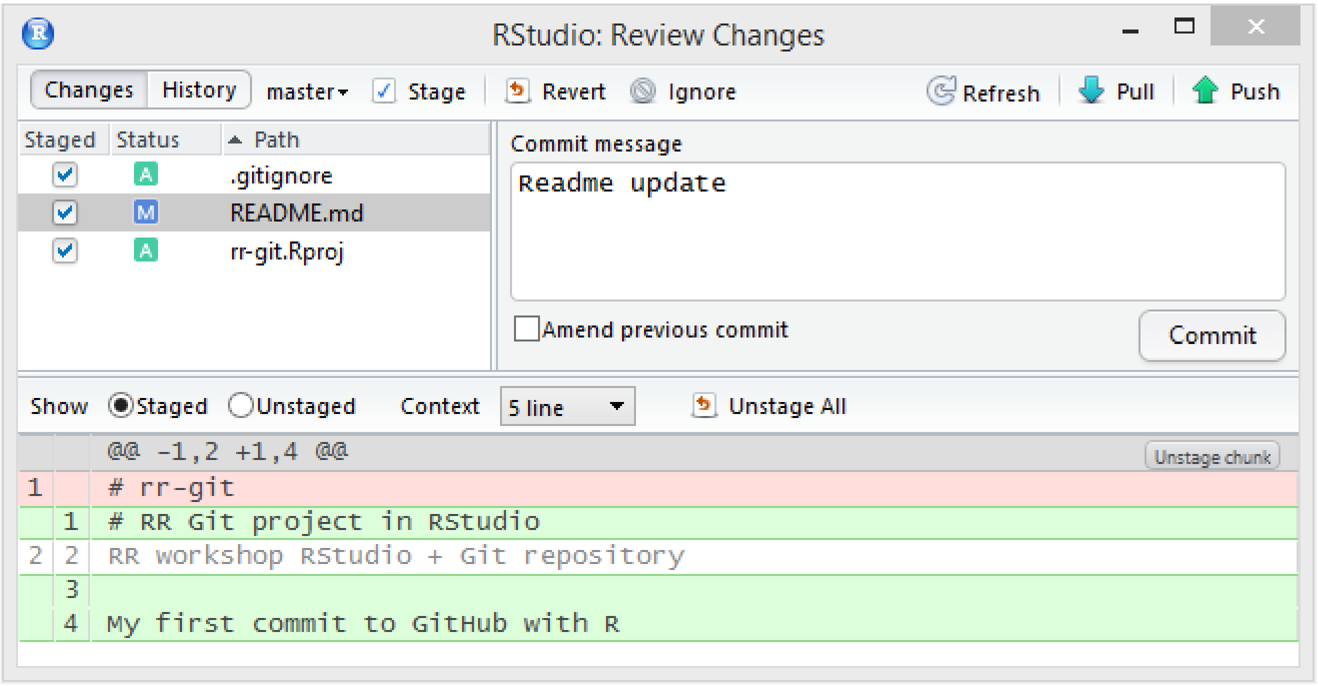
Part 3: Make local changes, commit and push to GitHub

1. Make local changes: Open the `README.md` file and edit and save the file.

```
# RR project in RStudio
RR workshop RStudio + Git repository

My first commit to GitHub with R
```

2. Commit the changes: Now we commit the local changes to the local Git repository.
In RStudio press the *Git* icon and select *Commit..* from Git menu (Ctrl+Alt+M) to open the commit window to review the changes in the repository. In the *Staged* column we select by checking the checkbox the files we want to commit. The lower pane shows the edits in green and red of the file. Enter a commit message to indicate what has changed in this commit e.g. `Readme update` and press the *Commit* button.



Staged	Status	Path
✓	A	.gitignore
✓	M	README.md
✓	A	rr-git.Rproj

Commit message

Readme update

```
@@ -1,2 +1,4 @@
1     # rr-git
   1  # RR Git project in RStudio
2  2  RR workshop RStudio + Git repository
   3
   4  My first commit to GitHub with R
```

3. Push to the remote repository: To push the changes to the remote GitHub repository press the *Push* button on the upper right corner of the commit window. You will be prompted to enter the username and password of your GitHub account. Enter them and check on the GitHub page if the changes got pushed to your online repository on GitHub.

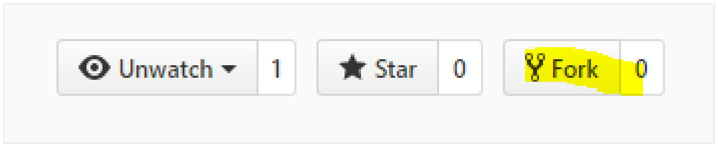
Now that you successfully pushed your first edits to a remote repository, repeat the above steps with a further file or R script that you create and edit, such as for instance the one below.

```
# Simple R file
# R example data.frame "cars"
str(cars)      # show the structure
summary(cars) # summary of the variables
plot(cars)     # plot speed against distance
```

Part 4: Fork a repository

Forking a project allows you to clone a repository on server-side and make it the starting point of your own project. A *fork* creates a personal copy of another repository. (See also the Github Forking guide)

1. Fork a repository on GitHub: Open https://github.com/314a/rr-r-publication and press the *fork* icon (in the upper right side of the project page) to fork this project to your own github account. On your GitHub page https://github.com/username the forked project should appear then in the list of your repositories.



2. Copy the HTTPS repository url: `https://github.com/yourusername/rr-r-publication.git` from *your* forked repository (see Part 2 Step 2) **3. Create a new RStudio project with Git** like you have already done in part 2 step 3 of this tutorial

Advanced: Link to a (different) remote repository

You may already have a local repository, like in this tutorial the `rstudio-git-test` repository, or want to fork your own project (which GitHub interestingly doesn't provide the options to do so). In this case you need to link your local repository with a (new) remote repository (and remove the old remote repository).

1. Create a new GitHub repository and copy the HTTPS url of the new repository e.g. `https://github.com/yourusername/rstudio-git-test2.git` .

2. Set the new remote repository in the Git shell: Open the Git shell from RStudio *Tools* > *Shell..* and type the following commands to set the new remote repository.
Type `git remote -v show` to show current remote repository to verify the location of the current repository.

```
git remote set-url origin https://github.com/yourusername/rstudio-git-test2.git
git remote add upstream https://github.com/yourusername/rstudio-git-test2.git
git push origin master
git push --all
```

Note: In case push/pull is greyed out in R stackoverflow use `git push -f origin master` and then `git push -u origin master` .

Note 2: Use `git remote rm origin` and `git remote rm upstream` if you want to remove the remote location from the current git folder. (origin is rather a convention than a command)

Further reading

[1] C. Brunsdon, L. Comber, An Introduction to R for Spatial Analysis & Mapping. London: Sage Publications Ltd, 2015.
[2] J. Paulson, "Version Control with Git and SVN," 2016. [Online].
Available: https://support.rstudio.com/hc/en-us/articles/200532077-Version-Control-with-Git-and-SVN.
[3] H. Wickham, "Git and GitHub," R packages, 2015. [Online]. Available: https://r-pkgs.had.co.nz/git.html.
[4] www.codeschool.com, "tryGit Tutorial." [Online]. Available: https://try.github.io. [5] K. Broman, "git/github guid." [Online]. Available: https://kbroman.org/github_tutorial/.

TODO check links for usefulness: https://rogerdudler.github.io/git-guide/ Terminology https://help.github.com/articles/github-glossary/ Practice tipps https://nvie.com/posts/a-successful-git-branching-model/ https://stat545-ubc.github.io/git03_rstudio-meet-git.htmlhttps://www.atlassian.com/git/tutorials/comparing-workflows/centralized-workflows