Pre-requisites for Git & GitHub (for the R user) Workshop

Francesca Vitalini

September 22, 2021

Introduction

Important Notice:

- **Course participants are expected to use their own laptop.**

This document contains the instructions to:

- install Git
- get an account on GitHub
- download and install R, RStudio and R packages that will be used during the workshop
- configure Git + GitHub + RStudio

These are *mandatory* preparation steps for the workshop.

Make sure to **do this well in advance** as there will not be the time for individual set up support during the workshop.

Install Git

Check existing installation

In the command line of the operating system, run:

- `which git` (`where git` on Windows) to get the git installation path
- `git --version` to check the git version you have installed

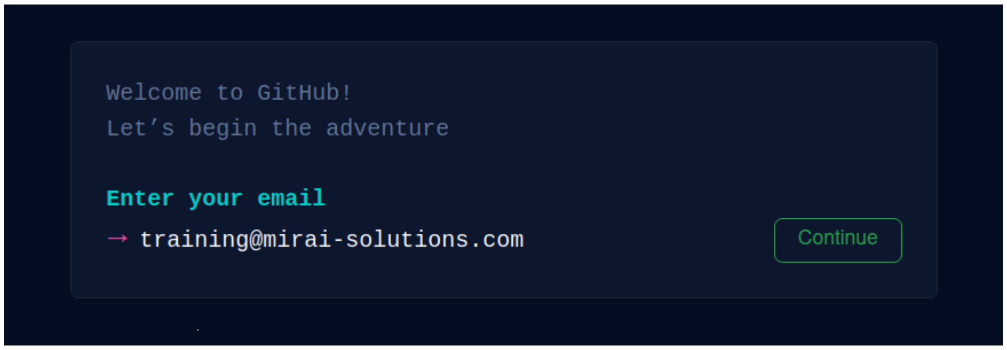
New installation

- on Windows, install Git from https://gitforwindows.org/
- on macOS, install Git from http://git-scm.com/downloads
- on Linux (Ubuntu), use the command `sudo apt install git`

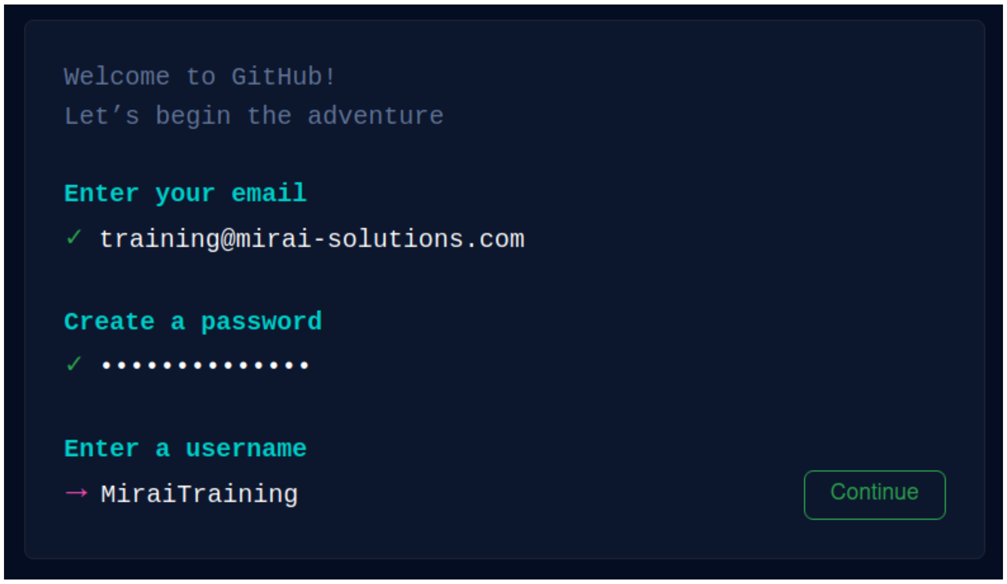
More Resources on installation: https://happygitwithr.com/install-git.html

Get a GitHub account

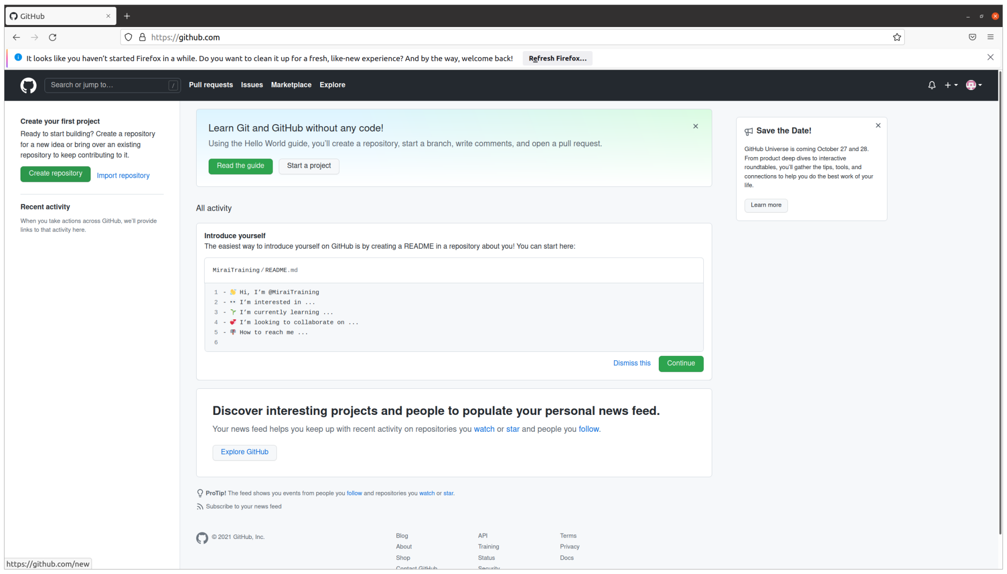
To follow along the hands-on part of the workshop you will need a GitHub account. If you do not have one already, sign up on GitHub with your email and username for free.



```
Welcome to GitHub!
Let's begin the adventure

Enter your email
→ training@mirai-solutions.com
```



```
Welcome to GitHub!
Let's begin the adventure

Enter your email
✓ training@mirai-solutions.com

Create a password
✓ ••••••••••••••

Enter a username
→ MiraiTraining
```



https://github.com

It looks like you haven't started Firefox in a while. Do you want to clean it up for a fresh, like-new experience? And by the way, welcome back!

Create your first project
Ready to start building? Create a repository for a new idea or bring over an existing repository to keep contributing to it.

Recent activity
When you take actions across GitHub, we'll provide links to that activity here.

Learn Git and GitHub without any code!
Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

All activity

Introduce yourself
The easiest way to introduce yourself on GitHub is by creating a README in a repository about you! You can start here:

MiraiTraining / README.md

```
- 👋 Hi, I'm @MiraiTraining
- 👀 I'm interested in ...
- 🌱 I'm currently learning ...
- 💞️ I'm looking to collaborate on ...
- 📫 How to reach me ...
```

Discover interesting projects and people to populate your personal news feed.
Your news feed helps you keep up with recent activity on repositories you watch or star and people you follow.

ProTip! The feed shows you events from people you follow and repositories you watch or star.

Subscribe to your news feed

Save the Date!
GitHub Universe is coming October 27 and 28. From product deep dives to interactive roundtables, you'll gather the tips, tools, and connections to help you do the best work of your life.

© 2021 GitHub, Inc.

https://github.com/new

Personalization may be skipped, or you can pick the options that you wish regarding your interests in using

GitHub.

Install R

This workshop illustrates Git and GitHub using R. To install the latest version of R (R version 4.1.1 (2021-08-10)) follow the instructions for your platform below.

R is a programming language and an environment for statistical computing and graphics.

R is a free, open source software and it is highly extensible through so-called *packages*, which can be freely downloaded from CRAN.

References:

- R: https://www.r-project.org
- CRAN: https://cran.r-project.org

UBUNTU:

Instructions for the installation of R 4.1 or later on Ubuntu 20.04. This should work also for 18.04 and 16.04.

Setup repository

From the command line of the operating system, configure the CRAN repository (run these lines as `root` or by prefixing `sudo`):

```
# update indices
apt update -qq
# install two helper packages we need
apt install --no-install-recommends software-properties-common dirmngr
# add the signing key (by Michael Rutter) for these repos
# To verify key, run gpg --show-keys /etc/apt/trusted.gpg.d/cran_ubuntu_key.asc
# Fingerprint: 298A3A825C0D65DFD57CBB651716619E084DAB9
wget -qO- https://cloud.r-project.org/bin/linux/ubuntu/marutter_pubkey.asc | sudo tee -a /etc/apt/truste
# add the R 4.0 repo from CRAN -- adjust 'focal' to 'groovy' or 'bionic' as needed
add-apt-repository "deb https://cloud.r-project.org/bin/linux/ubuntu $(lsb_release -cs)-cran40/"
```

Install R

From the command line of the operating system, install R by running

```
sudo apt update
sudo apt install r-base
```

To make sure that the `r-base-dev` package is installed, run

```
sudo apt install r-base-dev
```

References: * R installation: https://cran.r-project.org/bin/linux/ubuntu/

WINDOWS:

Download and execute the relevant installer from https://stat.ethz.ch/CRAN/bin/windows/base/

References:

- R installation https://stat.ethz.ch/CRAN/bin/windows/

MAC:

Download and execute the relevant installer from https://cran.r-project.org/bin/macosx/

Install RStudio

In this workshop we will code in R using RStudio.

RStudio is a free and open-source integrated development environment for the R programming language.

To install RStudio IDE, download the relevant installer from https://www.rstudio.com/products/rstudio/download/#download. On Ubuntu, the downloaded `rstudio-XXX.deb` is installed via

```
sudo dpkg -i rstudio-XXX.deb
```

References:

- RStudio: https://www.rstudio.com
- RStudio Cheat Sheets: https://www.rstudio.com/resources/cheatsheets/

Packages

In this workshop we will use a series of well-known packages. Make sure to have them pre-installed (`install.packages("<pkgName>")`).

Packages that we will use:

- `rmarkdown`
- `usethis`

E.g. `install.packages(c("usethis", "rmarkdown"))`

Configure Git + GitHub + RStudio

Set Git in RStudio

Set Git In RStudio: `Tools > Global Options > Git/SVN`

More Resources https://happygitwithr.com/rstudio-see-git.html

PAT

Password-based authentication for Git is deprecated and one should use a PAT (Private Access Token) instead.

Instructions to create a PAT on GitHub can be found here.

Otherwise use the helper function:

```
> usethis::create_github_token()
```

Select the desired options (default are fine for the scope of this workshop), then click on "Generate token" to create your PAT.

Remember to copy this token and store it somewhere, as once you leave the page you won't be able to see it again and would need to generate a new one.

However, **do not** put your PAT into your code! Same reason as why you would never put your password in your code, your PAT should be retrieved implicitly, for example, from the Git credential store or from an environment variable.

Store Credentials - Optional but Recommended

Store your credential in the Git Credential Store using command line in the terminal:

```
git config --global user.name '<your_github_username>'
git config --global user.email '<your_github_email>'
git config --global --get-regexp ^user
git config --global user.pat "<your github PAT>"
git config --global credential.helper store
```

Or in the R console you can use the `credentials` or `gitcreds` packages to access the Git Credentials Store. Here an example:

```
> usethis::use_git_config(user.name = "<your_github_username>",
+                         user.email = "<your_github_email>")
> credentials::set_github_pat() # You will be prompted to provide your GitHub PAT
> # credentials::git_credential_forget() to forget credentials
```

If you do not store the credentials you will be prompted to provide them at each git action.

To view the configuration, in the command line of the operating system:

```
git config --global --list
```

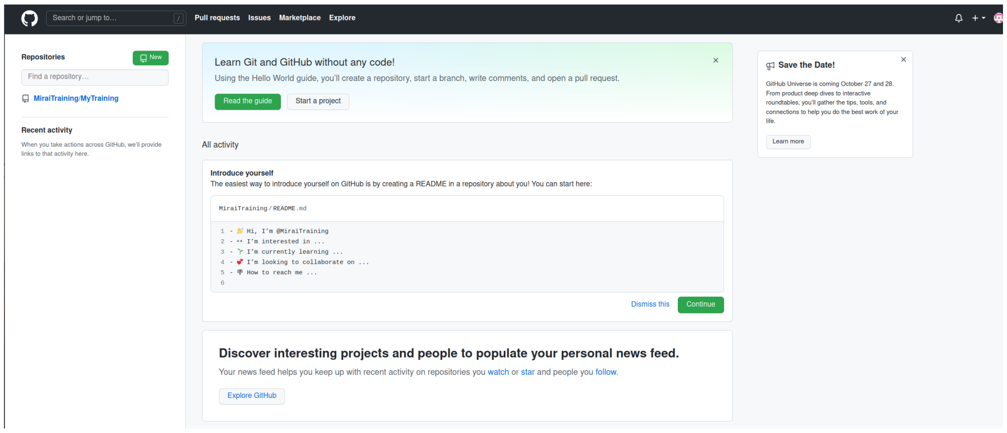
Test the Configuration

Due to the limited duration of the workshop there won't be any time during the workshop itself to fix personal installation / configuration issues. Therefore it is **extremely important** that you double-check that everything works well in advance.

Here a test you should run. If you have any questions / issues do not hesitate to get in touch!

1. Can you create a repo on GitHub?

- click on the "New" button on the top-left of your GitHub homepage (you need to be signed in first)



When you take actions across GitHub, we'll provide links to that activity here.

Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

The easiest way to introduce yourself on GitHub is by creating a README in a repository about you! You can start here:

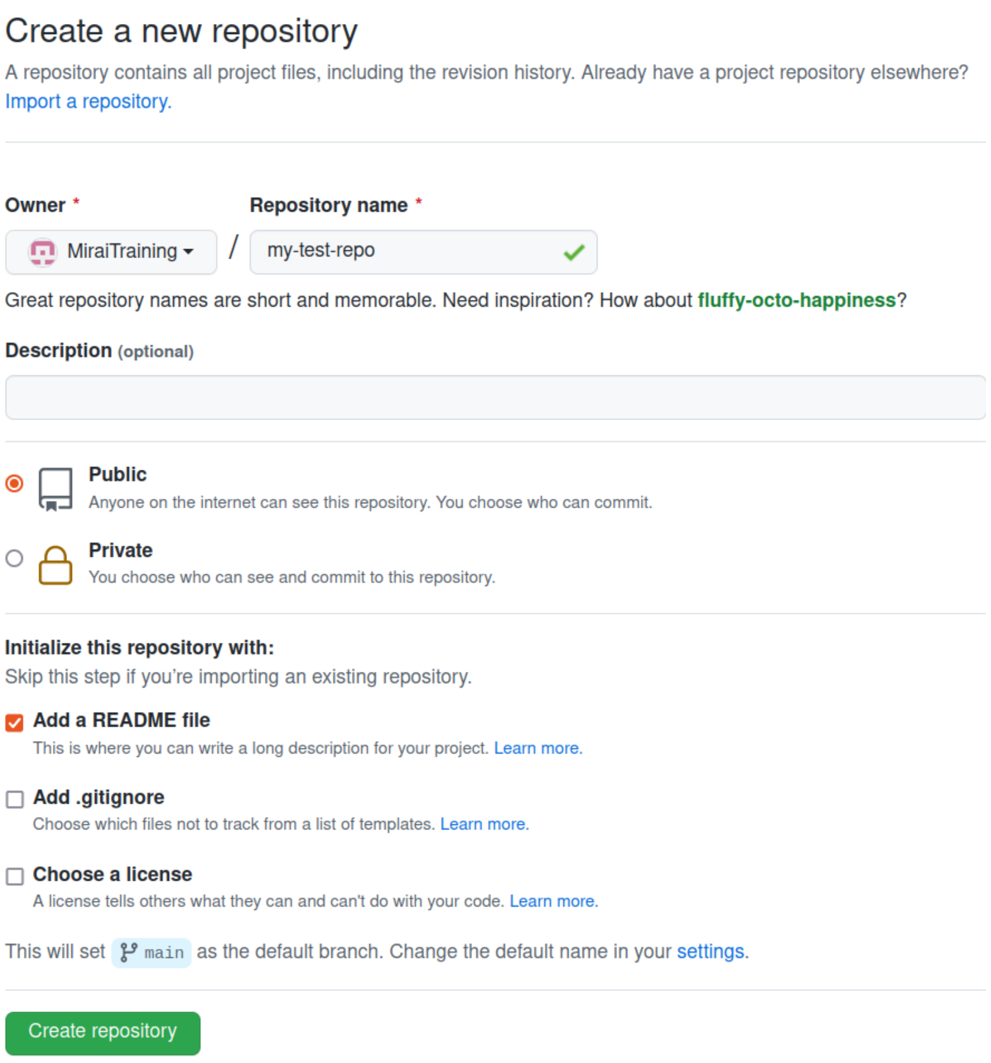
```
MiraiTraining / README.md

- 👋 Hi, I'm @MiraiTraining
- 👀 I'm interested in ...
- 🌱 I'm currently learning ...
- 💞 I'm looking to collaborate on ...
- 📫 How to reach me ...
```

GitHub Universe is coming October 27 and 28. From product deep dives to interactive roundtables, you'll gather the tips, tools, and connections to help you do the best work of your life.

Discover interesting projects and people to populate your personal news feed.

Your news feed helps you keep up with recent activity on repositories you watch or star and people you follow.

- give the new public repository a name and initialize it with a README file



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner * MiraiTraining / Repository name * my-test-repo

Great repository names are short and memorable. Need inspiration? How about fluffy-octo-happiness?

Description (optional)

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

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

Initialize this repository with:
Skip this step if you're importing an existing repository.

[✓ Add a README file] This is where you can write a long description for your project. Learn more.

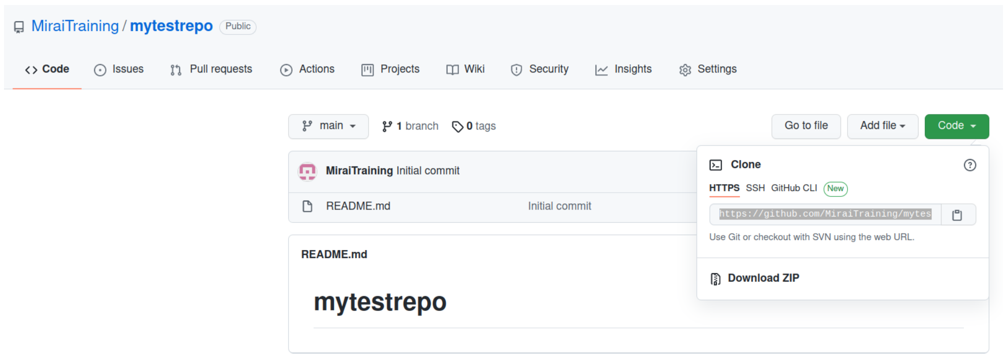
Add .gitignore Choose which files not to track from a list of templates. Learn more.

Choose a license A license tells others what they can and can't do with your code. Learn more.

This will set main as the default branch. Change the default name in your settings.

Create repository

- copy the GitHub url



MiraiTraining / mytestrepo Public

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 1 branch 0 tags Go to file Add file Code

MiraiTraining Initial commit

README.md Initial commit

Clone

HTTPS SSH GitHub CLI New

https://github.com/MiraiTraining/mytes

Use Git or checkout with SVN using the web URL.

Download ZIP

README.md

mytestrepo

2. From Command Line:

In the command line of the operating system:

- Get the test repo from GitHub:

```
git clone <your_github_url>
```

- Check what is in the test repo:

`cd <name_of_your_repo> ls head README.md git remote show origin`

- Add a line to the README: `echo "<Your Name> tested Git from command line on personal computer" >> README.md git status`
- Check that you can push changes to GitHub: `git add -A git status git commit -m "<Your Name> commit from command line" git push`
- Confirm you can see the changes on GitHub by visiting your repo on GitHub

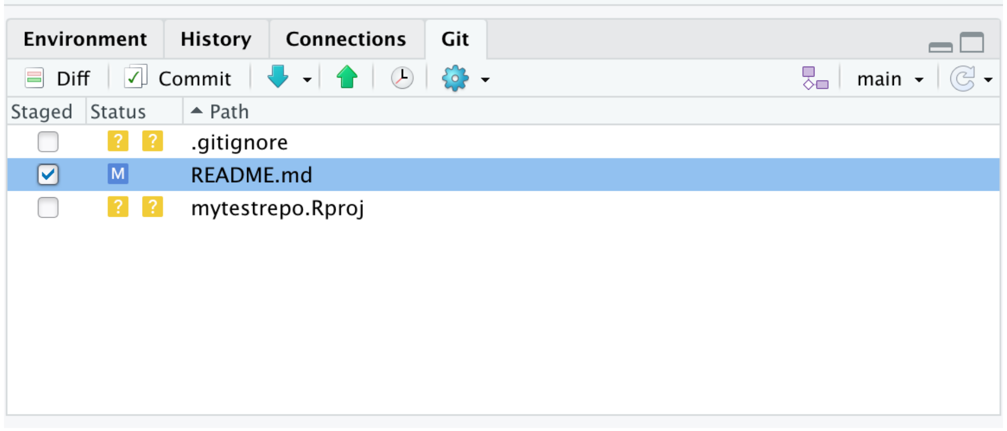
3. from RStudio:

In RStudio:

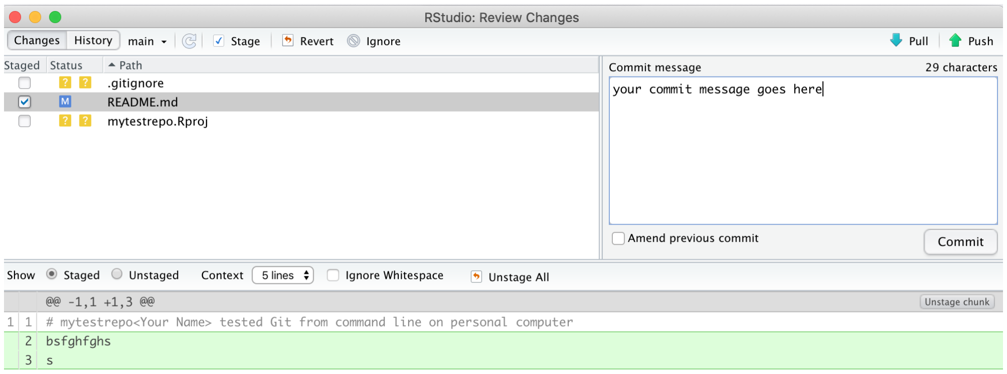
- Go to

`File > New Project... > Version Control > Git`

- Paste `<your_github_url>` as the repo URL in the the pop-up pane.
- Using everything as default, click on "Create Project".
- From RStudio, modify the `README.md` file by adding the line " tested Git from RStudio on personal computer". Save your changes.
- From the Git Pane in the top right of your RStudio:
  - Check box for `README.md`
  - in the pop-up, type the message: " commit from RStudio"
  - in the pop-up click "Commit"
  - in the pop-up click "Push"



Staged	Status	Path
☐	? ?	.gitignore
✓	M	README.md
☐	? ?	mytestrepo.Rproj



Staged	Status	Path
☐	? ?	.gitignore
✓	M	README.md
☐	? ?	mytestrepo.Rproj

Commit message 29 characters

```
your commit message goes here
```

```
@@ -1,1 +1,3 @@
# mytestrepo<Your Name> tested Git from command line on personal computer
bsfghfghs
s
```

- Confirm you can see the changes on GitHub by visiting your repo on GitHub

Final Checklist for the Workshop

After following these instructions we expect that you have successfully:

- obtained GitHub account
- installed/updated R and RStudio
- installed Git
- configured Git with your GitHub account
- confirmed that you can push to / pull from GitHub from the command line
- confirmed that you can push to / pull from GitHub from RStudio

If any of these steps did not work, please get in touch with us.

Looking forward to meeting you in the workshop "Git & GitHub (for the R user)".