

Git for R

Introduction and Manual¹

Nicolai Berk

University of Amsterdam

September 10, 2019

¹This presentation is based on: Happy Git with R

Why Git?

- "Microsoft Word's 'track changes' on steroids"
- Collaboration
 - Keep track of changes/versions (no tedious checking)
 - Share within team
 - Check each other's work
 - Others can contribute
- Public visibility

Basic features

- Issues
 - Discussions about changes tracked and linked to files
 - assign specific issues to persons
- Pull requests
 - Multiple independent branches
 - Merge back into Master or branch out

Preparation

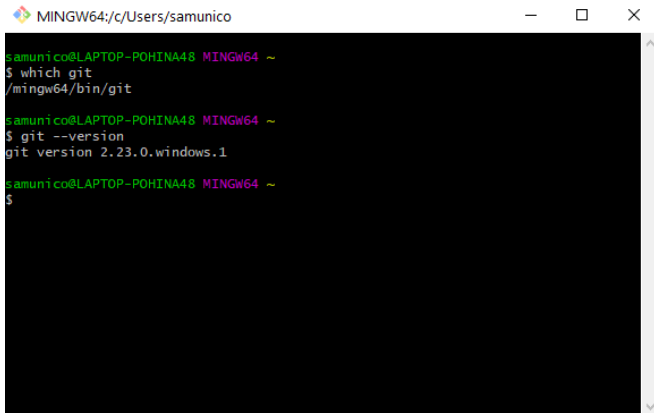
- Create a GitHub account: <https://github.com>
(preferably short and simple name)
- Update R: <https://cloud.r-project.org>
- Update RStudio:
<https://www.rstudio.com/products/rstudio/download>

⇒ This will ensure we're all on the same page

Get Git

- Install Git: <https://gitforwindows.org/>
 - Use default settings
 - **When asked about “Adjusting your PATH environment”, make sure to select “Git from the command line and also from 3rd-party software”**
- Check installation:
 - Open Git Bash Shell (came with installation of Git)
 - Type 'which git' and hit enter
 - Type 'git --version' and hit enter

Desired Output



```
MINGW64; c/Users/samunico
samunico@LAPTOP-POHINA48 MINGW64 ~
$ which git
/mingw64/bin/git

samunico@LAPTOP-POHINA48 MINGW64 ~
$ git --version
git version 2.23.0.windows.1

samunico@LAPTOP-POHINA48 MINGW64 ~
$
```

Figure: Your output should look somewhat like this.

Introduce yourself

- Open Git Bash Shell again
 - Type `'git config --global user.name '[your username]''`
 - Type `'git config --global user.email '[your email]''`
(upward arrow brings back last command)
 - test by typing `'git config --global --list'`
(check whether the named properties are correct)
- ! These properties should be the same as those registered in your GitHub profile earlier**

GUI: The RStudio of Git

As communicating with the command line can be a pain, and since GUI's might provide better overview, one should get a GUI. We use Sourcetree here.

⇒ get it here: <https://www.sourcetreeapp.com/>

Let's make a 'repo' !

- Go to <https://github.com> and make sure you are logged in.
- Click green “New repository” button.
 - Repository name: `myrepo`.
 - Description: “testing my setup”.
 - Public.
 - YES Initialize this repository with a README.
 - For everything else, just accept the default.
- Click big green button “Create repository.”
- Copy the HTTPS clone URL to your clipboard via the green “Clone or Download” button.

Clone the repo to your local computer

- Open the Git Bash Shell.
- Move around your directories using:

`pwd` show current working directory

`ls` show files in current directory

`cd [dir]` move into [dir]ectory

`cd ..` move one level upwards

`mkdir` make a new directory

- Find/create a folder of your choosing.
- Use the copied HTTPS one URL and type:
`git clone [URL]`

Clone output

```
samunico@LAPTOP-POHINA48 MINGW64 ~/OneDrive/Dokumente/GitHub/samunico/tmp
r)
$ git clone https://github.com/samunico/myrepo.git
Cloning into 'myrepo'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
```

Figure: Output should look like this.

Work in your repo

Make this new repo your working directory, list its files, display the README, and get some information on its connection to GitHub:

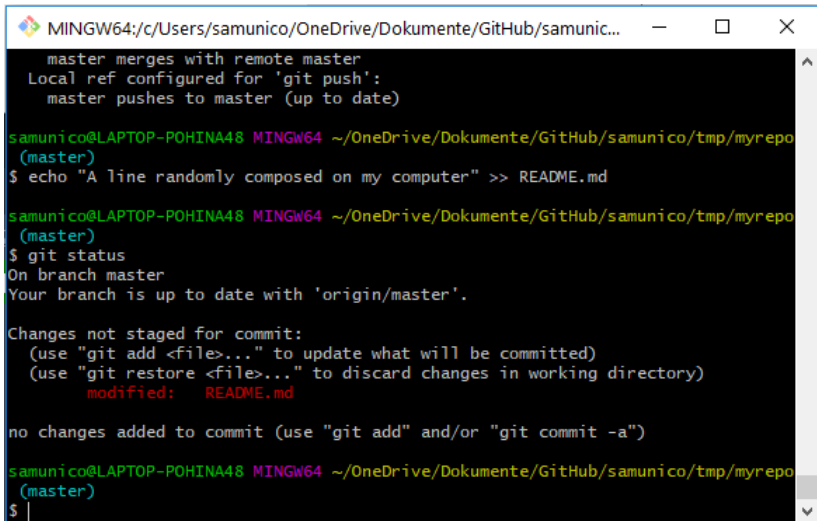
```
cd myrepo  
ls  
head README.md  
git remote show origin
```

Local change

Add a line to README and verify that Git notices the change:

```
echo "A line I wrote on my local computer" >>  
README.md  
git status
```

Status Output

A screenshot of a terminal window titled 'MINGW64:/c/Users/samunico/OneDrive/Dokumente/GitHub/samunic...'. The terminal shows the output of a 'git push' command, followed by a file creation and a 'git status' command. The status output indicates that the branch is up to date but there are uncommitted changes to 'README.md'.

```
master merges with remote master
Local ref configured for 'git push':
master pushes to master (up to date)

samunico@LAPTOP-POHINA48 MINGW64 ~/OneDrive/Dokumente/GitHub/samunico/tmp/myrepo
(master)
$ echo "A line randomly composed on my computer" >> README.md

samunico@LAPTOP-POHINA48 MINGW64 ~/OneDrive/Dokumente/GitHub/samunico/tmp/myrepo
(master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")

samunico@LAPTOP-POHINA48 MINGW64 ~/OneDrive/Dokumente/GitHub/samunico/tmp/myrepo
(master)
$ |
```

Figure: Output shows uncommitted, unstaged file.

Stage and commit

```
git add -A  
git commit -m "commit from my computer"  
git push
```

provide username and password when asked

-m "... " provides the commit message (states what this change is about)

Outcome

- go to github in your browser.
- refresh.
- README should have a new line.
- When you click on 'commits', you should see the commit message

Cleaning

local Delete the repository using `rm -rf myrepo`.

GitHub Go to your repo's landing page on GitHub. Click on "Settings". Scroll down, click on "delete repository," and do as it asks.

ConnectR

- On `github.com`, create a new empty repository as you did before.
- Open a new project in RStudio via *File ↵ New Project ↵ Version Control ↵ Git*.
- Repository URL = Clone with HTTPS
- Accept defaults.
- Create Project.

Make changes in RStudio

- Open README.md by clicking on it.
- Add another line.
- Commit:
 - Click 'Git' in the toolbar.
 - Commit.
 - Check box 'staged' for README.md.
 - Type commit message.
 - Commit.
 - Click the green 'push' button in the upper right corner.
 - Go to GitHub and check whether the changes have been committed.

Congratulations

YOU NOW HAVE THE ABILITY TO COMMIT CHANGES
DIRECTLY FROM RSTUDIO!!!!