Git and GitLab/GitHub

A Basic Guide to Version Control and Collaboration

Luke Hannan

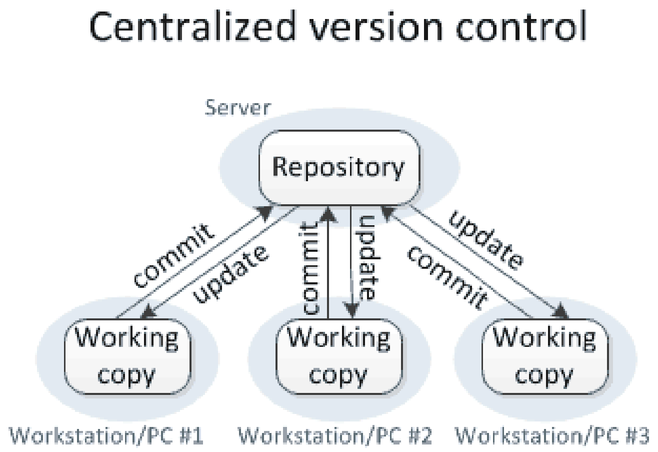
20 February 2021

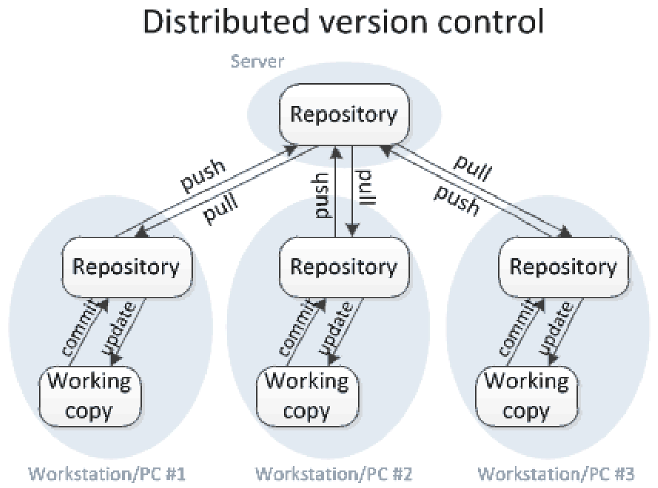
Outline

About Git

Git is a Distributed Version Control System (DVCS) designed to track changes in code.

It was designed to allow multiple collaborators to work on the same project; and to allow all collaborators to have the entire project's history on their local device without connecting to a central server.

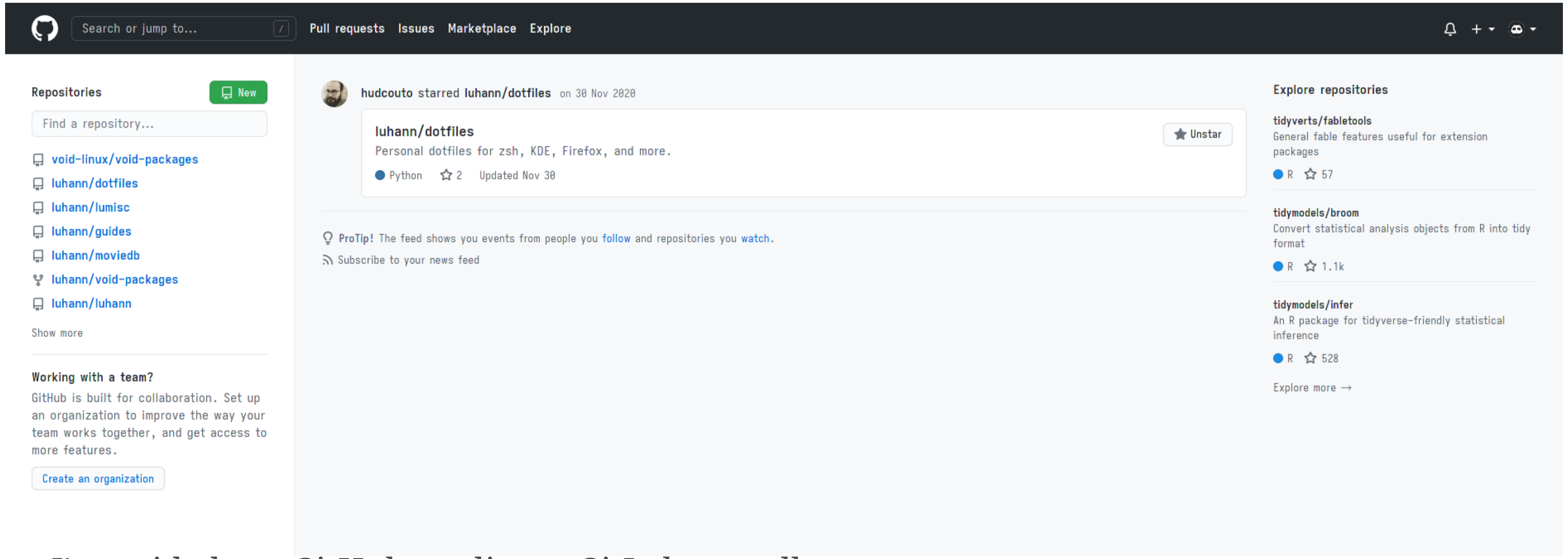




Source: https://homes.cs.washington.edu/~mernst/advice/version-control.html

About GitHub and GitLab

- GitHub is a **web-based** service for storing, sharing, and collaborating on projects managed with Git.
- GitHub makes collaborating with other people much easier than using git alone.
- GitHub also provides an easy to use website for collaboration.



void-linux/void-packages
luhann/dotfiles
luhann/lumisc
luhann/guides
luhann/moviedb
luhann/void-packages
luhann/luhann

Working with a team?
GitHub is built for collaboration. Set up an organization to improve the way your team works together, and get access to more features.

hudcouto starred luhann/dotfiles on 30 Nov 2020

luhann/dotfiles
Personal dotfiles for zsh, KDE, Firefox, and more.
Python 2 Updated Nov 30

ProTip! The feed shows you events from people you follow and repositories you watch.
Subscribe to your news feed

tidyverts/fabletools
General fable features useful for extension packages
R 57

tidymodels/broom
Convert statistical analysis objects from R into tidy format
R 1.1k

tidymodels/infer
An R package for tidyverse-friendly statistical inference
R 528

* Most of what I've said about GitHub applies to GitLab as well.

Why use Git?

- **Fast**: No network overhead, unlike centralized version control software.
- **Distributed**: Each user has the entire project history on their local machine.
- **Local**: You don't need internet to use it. You only need internet if you want to share your code with others (e.g., using GitLab).
- **Branches**: Keep your coding experiments separate from code that is already working.

Why use Git for *Research*?

Git is useful for researchers in a number ways. It can help:

- Protect yourself from accidental deletion:

```
# It's easy to make mistakes like these
rm -rf *
rm -rf ./
```

- If you accidentally **overwrote** important code to test something new.

- Perhaps you copied your code file to test new things:

```
# This isn't necessarily bad, but it's easy to lose track of all the changes that you make
cp working_code.R new_function_test.R
```

- Finally, Git also helps others to collaborate on your **code**.

Key Words to Understanding Git

Key Terminology for using Git and GitLab*

Repository is where you keep all the files you want to track. The convention is usually create a repository per project (aka repo).

Commit is an object that holds information about a particular change. Think of **commits** as the snapshots of your work.

HEAD refers to the most recent commit on the current branch.

Branch is the name for a separate line of development, with its own history.

***** We will go over these terms again later.

Key Terminology for using Git and GitLab*

Pull: When using an online repo (like GitLab), Pull will download all changes from the online repo and automatically merge them into your current project.

Push: The opposite of pull, git push will transfer all changes you made on your machine to the online repo.

Fork: Forking is a GitLab thing (not git) and allows you to copy a repo to your account and make changes without it affecting the original.

Merge: Copy your fork back to the original repo and add your changes to it.

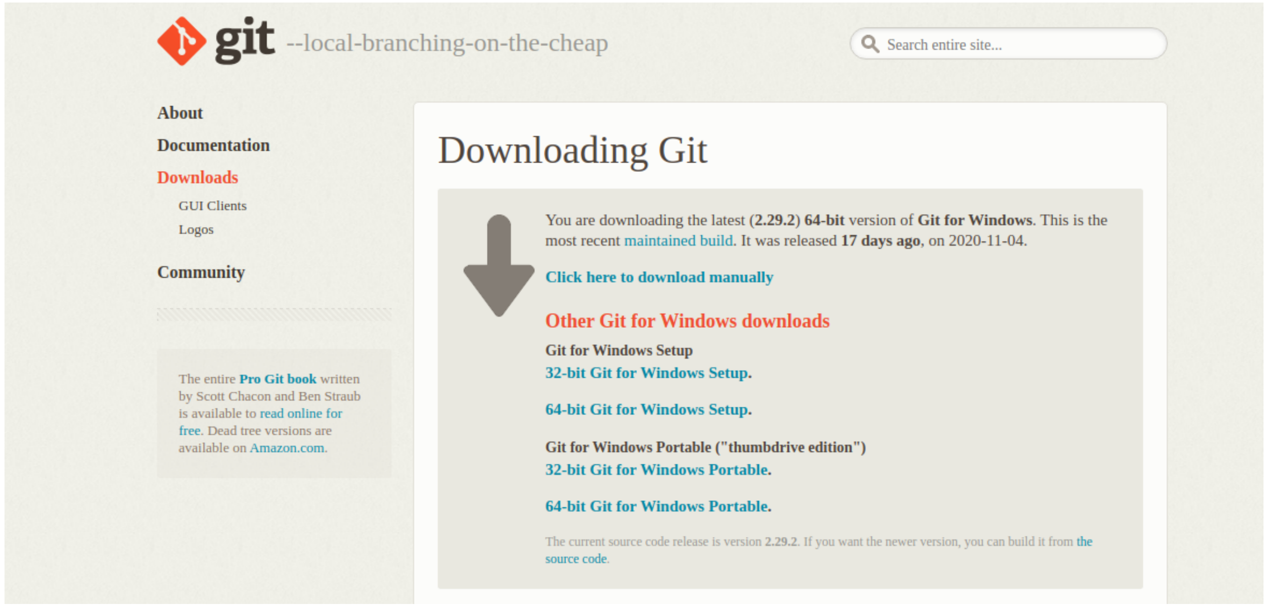
***** All of the terms in this slide refer to GitHub/GitLab terms and can be ignored if you are not using them.

How to install Git

How to install Git

On Windows

- Go to the windows download page on the Git website
- Download the version of Git you want:



The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

Downloading Git

You are downloading the latest (**2.29.2**) **64-bit** version of **Git for Windows**. This is the most recent maintained build. It was released **17 days ago**, on 2020-11-04.

Click here to download manually

Other Git for Windows downloads

Git for Windows Setup
32-bit Git for Windows Setup.

64-bit Git for Windows Setup.

Git for Windows Portable ("thumbdrive edition")
32-bit Git for Windows Portable.

64-bit Git for Windows Portable.

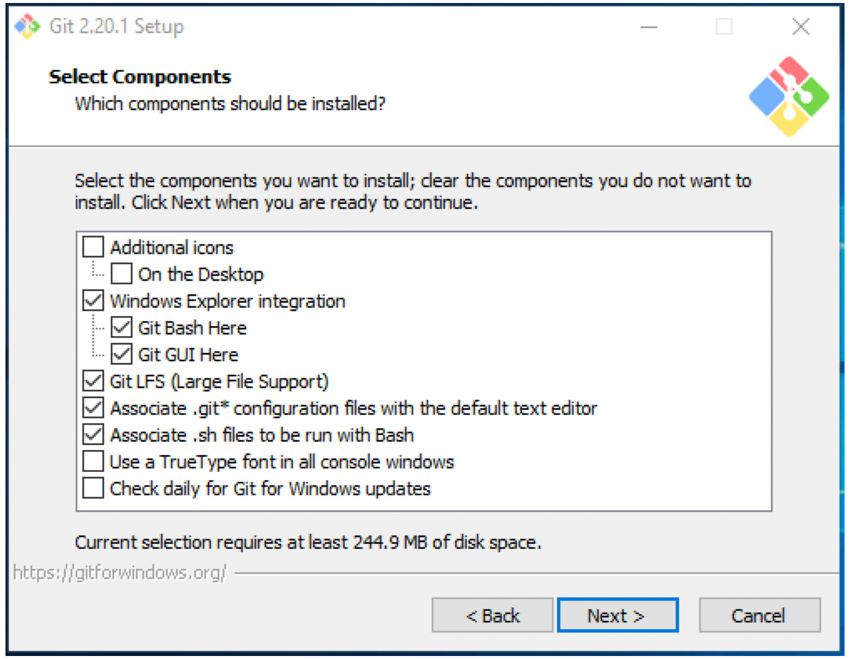
The current source code release is version **2.29.2**. If you want the newer version, you can build it from the source code.

Source: https://www.linode.com/docs/guides/how-to-install-git-on-linux-mac-and-windows/

How to install Git

On Windows

- After running installer, select the components you want to install (if you are unsure, go ahead with the default selection):



Git 2.20.1 Setup

Select Components

Which components should be installed?

Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- [ ] Additional icons
  - [ ] On the Desktop
- [x] Windows Explorer integration
  - [x] Git Bash Here
  - [x] Git GUI Here
- [x] Git LFS (Large File Support)
- [x] Associate .git* configuration files with the default text editor
- [x] Associate .sh files to be run with Bash
- [ ] Use a TrueType font in all console windows
- [ ] Check daily for Git for Windows updates

Current selection requires at least 244.9 MB of disk space.

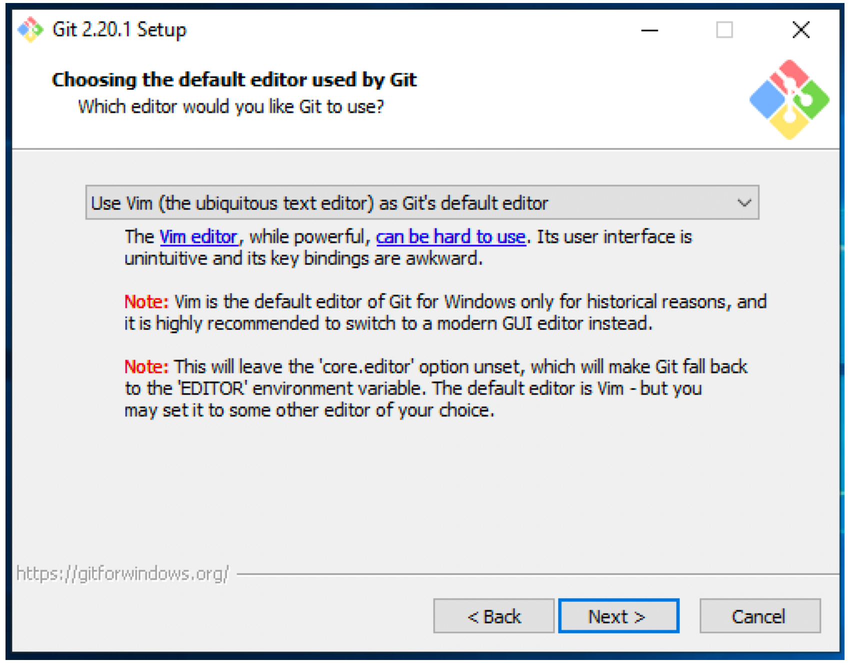
https://gitforwindows.org/

Source: https://www.linode.com/docs/guides/how-to-install-git-on-linux-mac-and-windows/

How to install Git

On Windows

- Choose the default editor for Git, I recommend you change this to your editor of choice:



Git 2.20.1 Setup

Choosing the default editor used by Git

Which editor would you like Git to use?

Use Vim (the ubiquitous text editor) as Git's default editor

The Vim editor, while powerful, can be hard to use. Its user interface is unintuitive and its key bindings are awkward.

Note: Vim is the default editor of Git for Windows only for historical reasons, and it is highly recommended to switch to a modern GUI editor instead.

Note: This will leave the 'core.editor' option unset, which will make Git fall back to the 'EDITOR' environment variable. The default editor is Vim - but you may set it to some other editor of your choice.

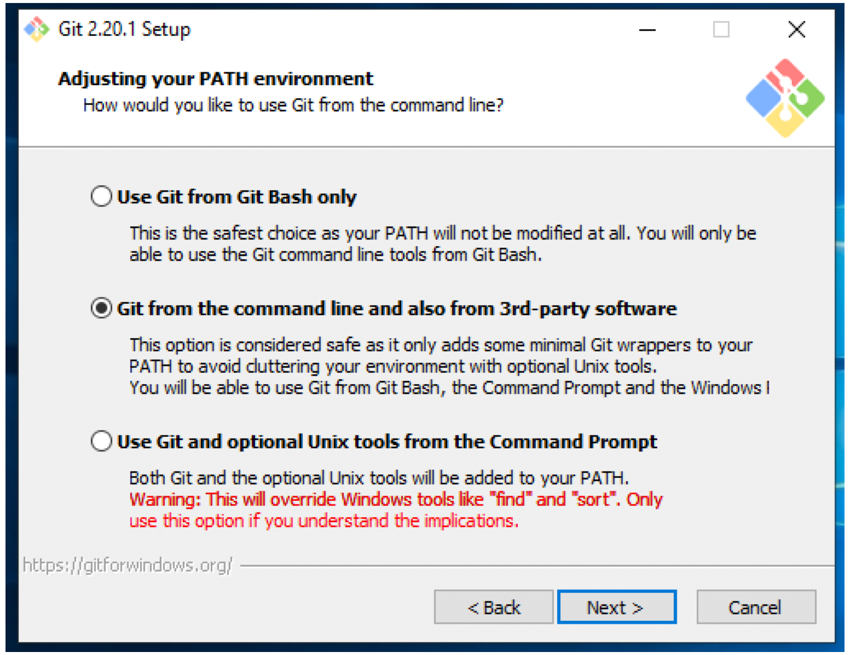
https://gitforwindows.org/

Source: https://www.linode.com/docs/guides/how-to-install-git-on-linux-mac-and-windows/

How to install Git

On Windows

- Choose how to use Git, the middle option is the safest:



Git 2.20.1 Setup

Adjusting your PATH environment

How would you like to use Git from the command line?

( ) **Use Git from Git Bash only**

This is the safest choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.

(●) **Git from the command line and also from 3rd-party software**

This option is considered safe as it only adds some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools.
You will be able to use Git from Git Bash, the Command Prompt and the Windows I

( ) **Use Git and optional Unix tools from the Command Prompt**

Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.

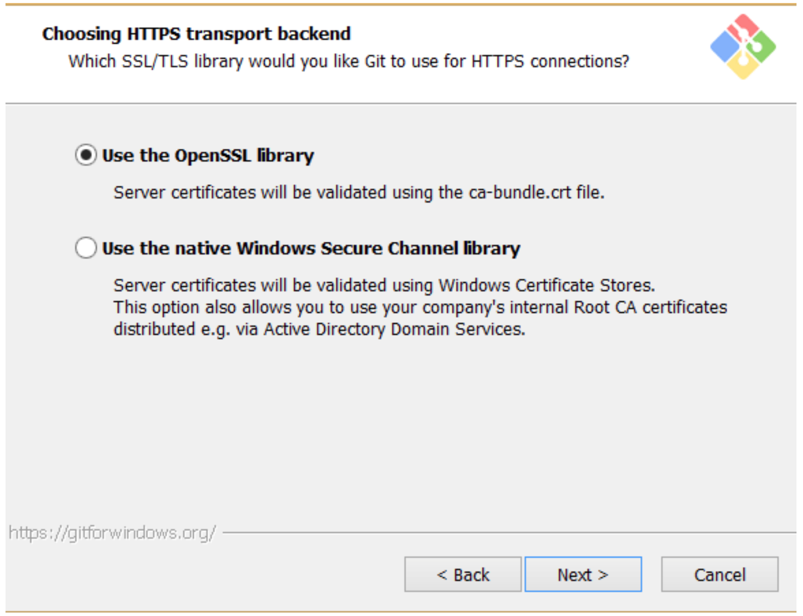
https://gitforwindows.org/

Source: https://www.linode.com/docs/guides/how-to-install-git-on-linux-mac-and-windows/

How to install Git

On Windows

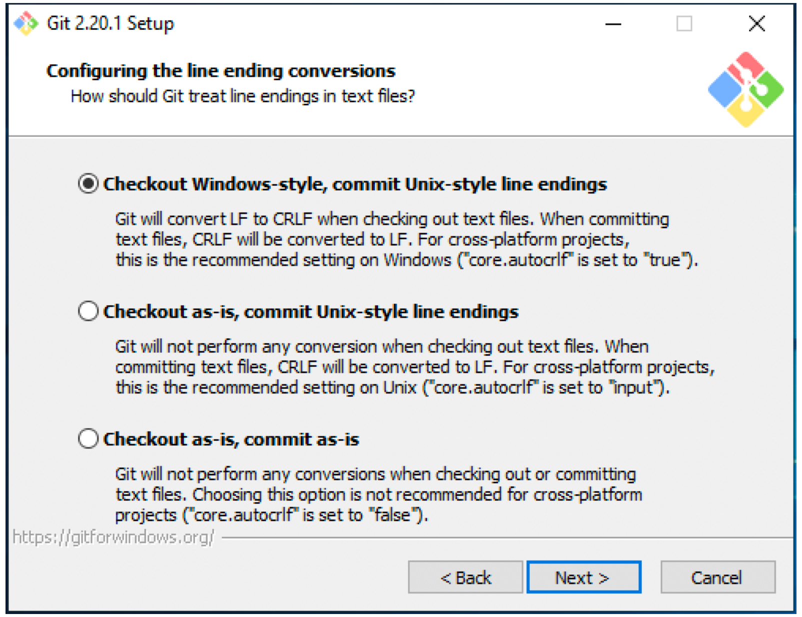
- The rest of the default options are fine to select:



Choosing HTTPS transport backend
Which SSL/TLS library would you like Git to use for HTTPS connections?

[✓] **Use the OpenSSL library**
Server certificates will be validated using the ca-bundle.crt file.

[ ] **Use the native Windows Secure Channel library**
Server certificates will be validated using Windows Certificate Stores.
This option also allows you to use your company's internal Root CA certificates distributed e.g. via Active Directory Domain Services.



Git 2.20.1 Setup

Configuring the line ending conversions
How should Git treat line endings in text files?

[✓] **Checkout Windows-style, commit Unix-style line endings**
Git will convert LF to CRLF when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Windows ("core.autocrlf" is set to "true").

[ ] **Checkout as-is, commit Unix-style line endings**
Git will not perform any conversion when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Unix ("core.autocrlf" is set to "input").

[ ] **Checkout as-is, commit as-is**
Git will not perform any conversions when checking out or committing text files. Choosing this option is not recommended for cross-platform projects ("core.autocrlf" is set to "false").

Source: https://www.linode.com/docs/guides/how-to-install-git-on-linux-mac-and-windows/

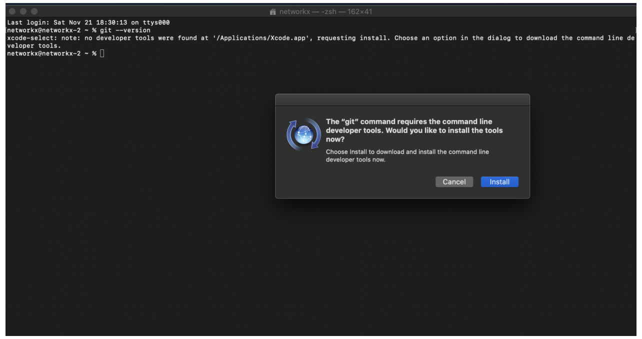
How to install Git

On Mac

- Check if Git is already installed on your Mac using:

```
# In terminal type
git --version
```

- If Git isn't installed on your computer, the terminal prompts you with the following message:



```
Last login: Sat Nov 21 18:30:13 on ttys000
networkx@networkx-2 ~ % git --version
xcode-select: note: no developer tools were found at '/Applications/Xcode.app', requesting install. Choose an option in the dialog to download the command line de
veloper tools.
networkx@networkx-2 ~ % 
```

The "git" command requires the command line developer tools. Would you like to install the tools now?

Choose Install to download and install the command line developer tools now.

- Click on Install

Source: https://www.linode.com/docs/guides/how-to-install-git-on-linux-mac-and-windows/

How to install Git

On Linux (Ubuntu)

- In a terminal, update first:

```
# In terminal type
sudo apt update
```

- Then install Git:

```
# In terminal type
sudo apt install git
```

- Check if Git installed correctly:

```
# In terminal type
git --version
```

How to use Git

How to use Git and GitLab when you are collaborating with others

Setting Up Git

This is only done once

- Git needs to know who you are to assign authorship to your commits.

```
# Tell git your name
git config --global user.name "Dr. Henry Jekyll"

# Tell git your email
git config --global user.email jekyll@example.com
```

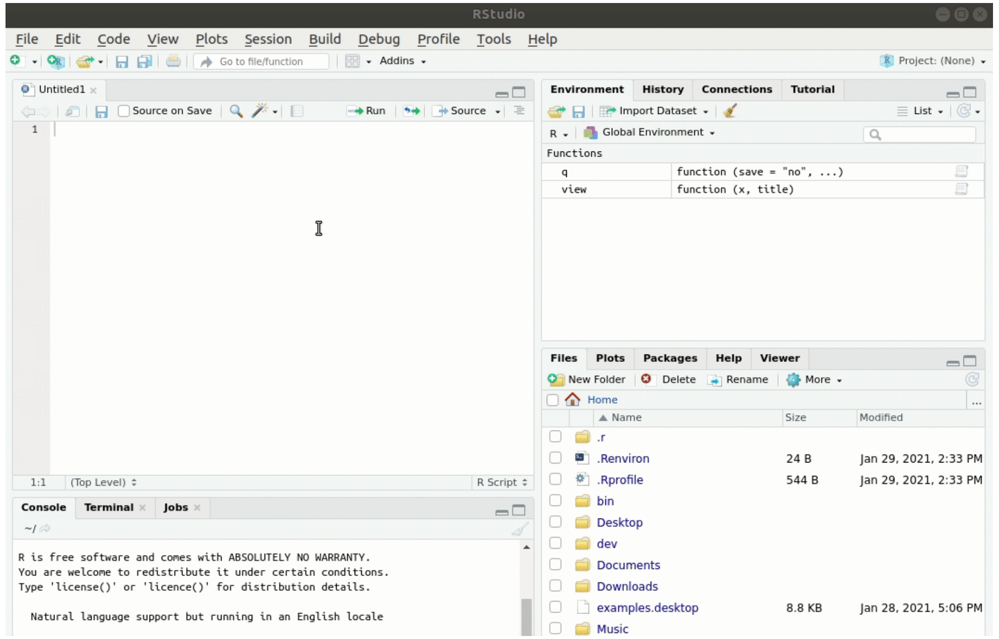
Typical Git Workflow

Creating a Repository

- First we initialize the repository. This creates an empty repository that allows us to add files and folders to be tracked by git.

```
# Move to your project directory
cd new_project
# Initialize repository
# This is only done once at the start
# of each project
git init .

# At any point you can see that status of
# your git repository
git status
```


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Untitled1
Source on Save
Run
Source
1:1
(Top Level)
R Script
Environment History Connections Tutorial
Import Dataset
List
Global Environment
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home
Console Terminal Jobs

Functions

q	function (save = "no", ...)
view	function (x, title)

Name	Size	Modified
.r		
.Renviron	24 B	Jan 29, 2021, 2:33 PM
.Rprofile	544 B	Jan 29, 2021, 2:33 PM
bin		
Desktop		
dev		
Documents		
Downloads		
examples.desktop	8.8 KB	Jan 28, 2021, 5:06 PM
Music		

```
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale
```

Typical Git Workflow

Adding Files

- Once we have created our new repo, we need to tell git which files to track.

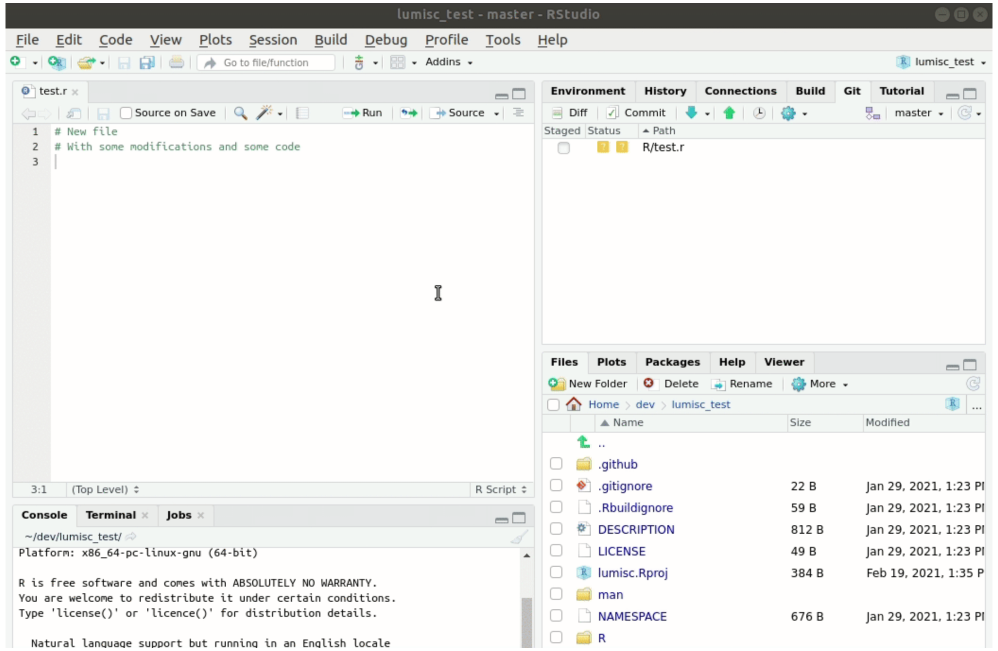
```
# In your project directory
git add R/test.r

# Adding files "stages" them telling git
# we want to track them

# See our added files
git status

# "staged" files need to be "committed" for
# the changes to be permanent
git commit R/test.r

# You can commit files with a message as well
git commit R/test.r --message "An informative
```



lumisc_test - master - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
lumisc_test
test.r
Source on Save
Run
Source
3:1
(Top Level)
R Script
Console Terminal Jobs
Environment History Connections Build Git Tutorial
Diff Commit
master
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home > dev > lumisc_test

```
# New file
# With some modifications and some code
```

```
~/dev/lumisc_test/
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale
```

Staged	Status	Path
☐	? ?	R/test.r

Name	Size	Modified
..		
.github		
.gitignore	22 B	Jan 29, 2021, 1:23 PI
.Rbuildignore	59 B	Jan 29, 2021, 1:23 PI
DESCRIPTION	812 B	Jan 29, 2021, 1:23 PI
LICENSE	49 B	Jan 29, 2021, 1:23 PI
lumisc.Rproj	384 B	Feb 19, 2021, 1:35 P
man		
NAMESPACE	676 B	Jan 29, 2021, 1:23 PI
R		

Typical Git Workflow

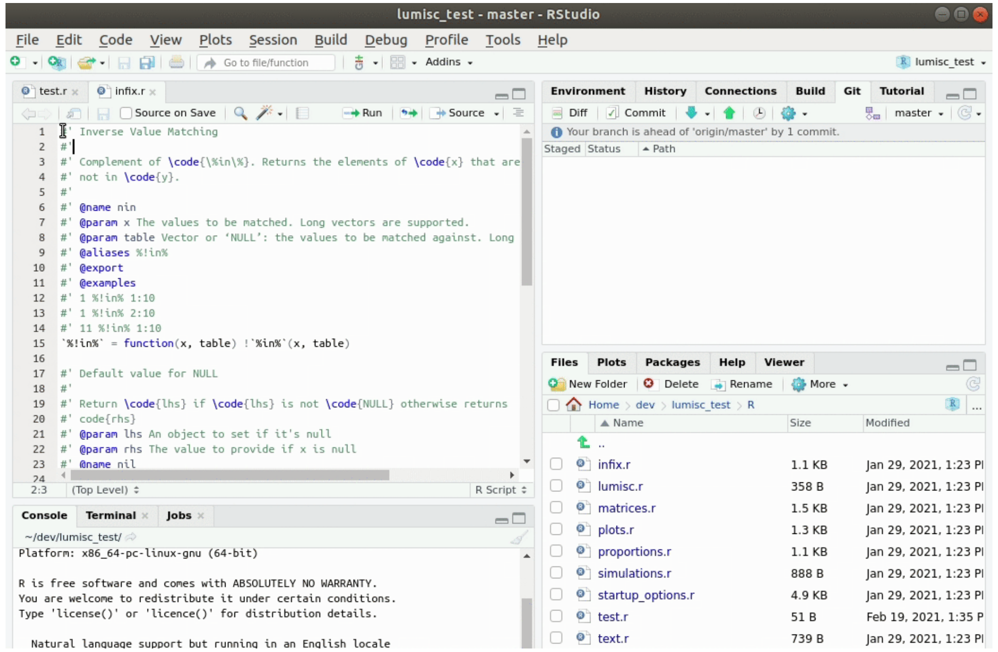
Changing Existing Files

- Once files have been added and commited, we can make changes that are permanently tracked.

```
# First edit or change Files

# See what files have changed
git status

# Commit our newly changed files
git commit R/infix.r
# or with a message
git commit R/infix.r --message "We changed an
```



lumisc_test - master - RStudio

Your branch is ahead of 'origin/master' by 1 commit.

```
#' Inverse Value Matching
#'
#' Complement of \code{\%in\%}. Returns the elements of \code{x} that are
#' not in \code{y}.
#'
#' @name nin
#' @param x The values to be matched. Long vectors are supported.
#' @param table Vector or ‘NULL’: the values to be matched against. Long
#' @aliases %!in%
#' @export
#' @examples
#' 1 %!in% 1:10
#' 1 %!in% 2:10
#' 11 %!in% 1:10
`%!in%` = function(x, table) !`%in%`(x, table)

#' Default value for NULL
#'
#' Return \code{lhs} if \code{lhs} is not \code{NULL} otherwise returns
#' code{rhs}
#' @param lhs An object to set if it's null
#' @param rhs The value to provide if x is null
#' @name nil
```

```
~/dev/lumisc_test/
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale
```

Home › dev › lumisc_test › R

Name	Size	Modified
..		
infix.r	1.1 KB	Jan 29, 2021, 1:23 PI
lumisc.r	358 B	Jan 29, 2021, 1:23 PI
matrices.r	1.5 KB	Jan 29, 2021, 1:23 PI
plots.r	1.3 KB	Jan 29, 2021, 1:23 PI
proportions.r	1.1 KB	Jan 29, 2021, 1:23 PI
simulations.r	888 B	Jan 29, 2021, 1:23 PI
startup_options.r	4.9 KB	Jan 29, 2021, 1:23 PI
test.r	51 B	Feb 19, 2021, 1:35 P
text.r	739 B	Jan 29, 2021, 1:23 PI

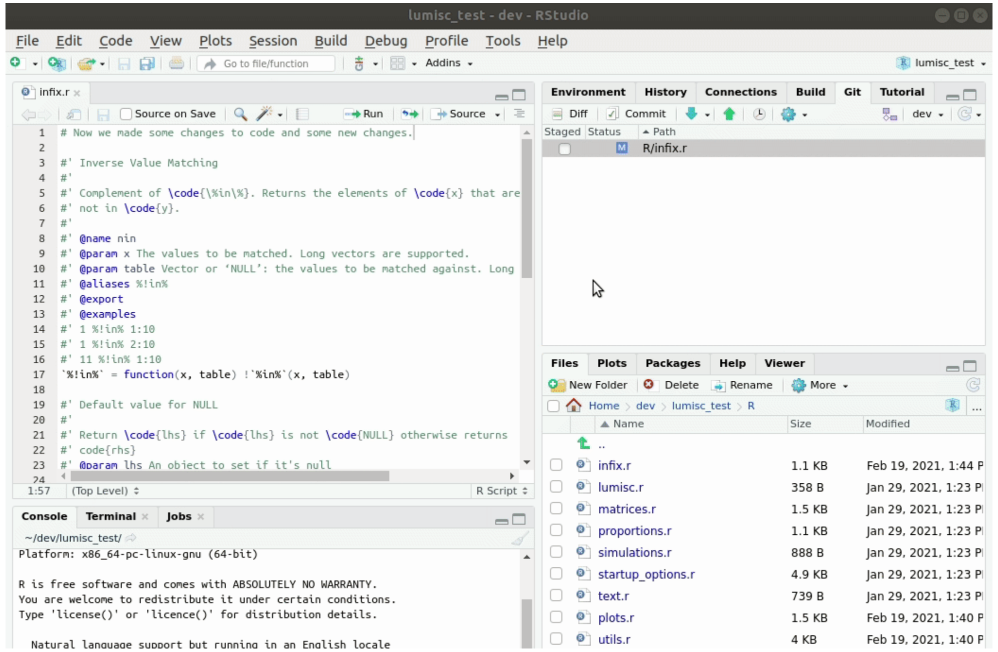
Typical Git Workflow

Reverting Changes

- Perhaps you have **staged** some changes but you don't want them anymore and want to revert the file to it's original state.

```
# See any potentially changed Files
git status

# Revert changes to latest commit
git revert R/infix.r
```


lumisc_test - dev - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
lumisc_test
infix.r
Source on Save
Run
Source
Environment History Connections Build Git Tutorial
Diff Commit
dev
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home › dev › lumisc_test › R
Console Terminal Jobs

```
# Now we made some changes to code and some new changes.

#' Inverse Value Matching
#'
#' Complement of \code{\%in\%}. Returns the elements of \code{x} that are
#' not in \code{y}.
#'
#' @name nin
#' @param x The values to be matched. Long vectors are supported.
#' @param table Vector or ‘NULL’: the values to be matched against. Long
#' @aliases %!in%
#' @export
#' @examples
#' 1 %!in% 1:10
#' 1 %!in% 2:10
#' 11 %!in% 1:10
`%!in%` = function(x, table) !`%in%`(x, table)

#' Default value for NULL
#'
#' Return \code{lhs} if \code{lhs} is not \code{NULL} otherwise returns
#' code{rhs}
#' @param lhs An object to set if it's null
```

1:57 (Top Level) R Script

Staged	Status	Path
	M	R/infix.r

```
~/dev/lumisc_test/
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale
```

Name	Size	Modified
..		
infix.r	1.1 KB	Feb 19, 2021, 1:44 P
lumisc.r	358 B	Jan 29, 2021, 1:23 P
matrices.r	1.5 KB	Jan 29, 2021, 1:23 P
proportions.r	1.1 KB	Jan 29, 2021, 1:23 P
simulations.r	888 B	Jan 29, 2021, 1:23 P
startup_options.r	4.9 KB	Jan 29, 2021, 1:23 P
text.r	739 B	Jan 29, 2021, 1:23 P
plots.r	1.5 KB	Feb 19, 2021, 1:40 P
utils.r	4 KB	Feb 19, 2021, 1:40 P

Typical Git Workflow

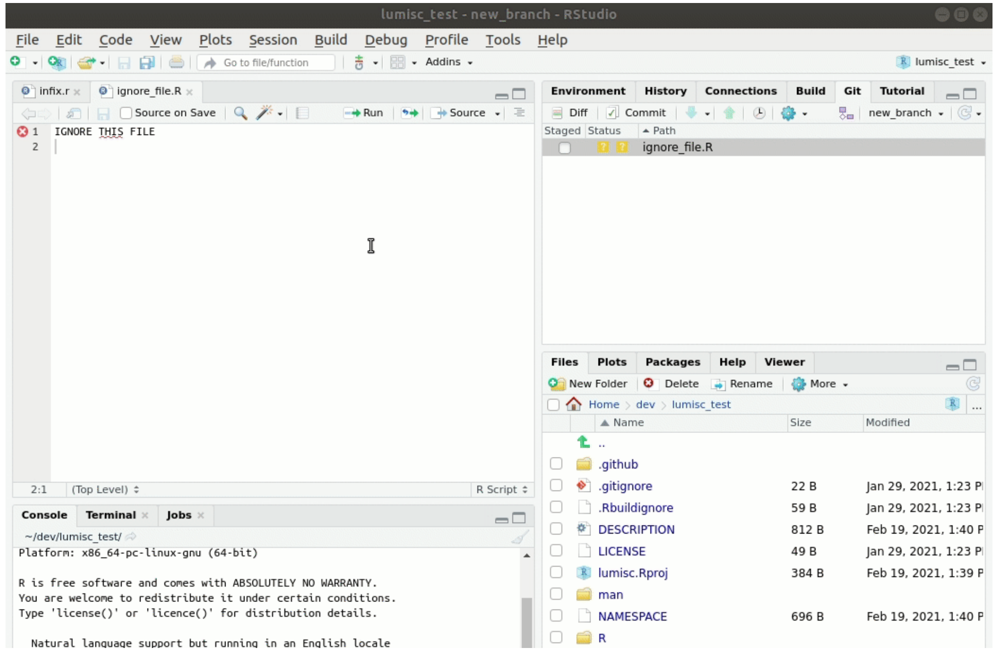
Ignoring Certain Files

- Some files in your project will either be too unwieldy (very large datasets) or not necessary (output files generated by code) for Git to track, but we can tell git to ignore these files with a special file the **.gitignore** file. Ignored files will not be tracked by git at all.

```
# See any potentially changed Files
git status

# Make the .gitignore file in any way
# you choose
echo "ignore_file.R" >> .gitignore

# Make sure to commit
# the .gitignore file
git add .gitignore
git commit .gitignore --message "Add gitignor
```



lumisc_test - new_branch - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
lumisc_test
infix.r
ignore_file.R
Source on Save
Run
Source
Environment History Connections Build Git Tutorial
Diff Commit
new_branch
Staged Status Path
2:1 (Top Level) R Script
Console Terminal Jobs
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home dev lumisc_test

```
IGNORE THIS FILE
```

Staged	Status	Path
		ignore_file.R

```
~/dev/lumisc_test/
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale
```

Name	Size	Modified
..		
.github		
.gitignore	22 B	Jan 29, 2021, 1:23 P
.Rbuildignore	59 B	Jan 29, 2021, 1:23 P
DESCRIPTION	812 B	Feb 19, 2021, 1:40 P
LICENSE	49 B	Jan 29, 2021, 1:23 P
lumisc.Rproj	384 B	Feb 19, 2021, 1:39 P
man		
NAMESPACE	696 B	Feb 19, 2021, 1:40 P
R		

Typical Git Workflow

Creating a New Branch

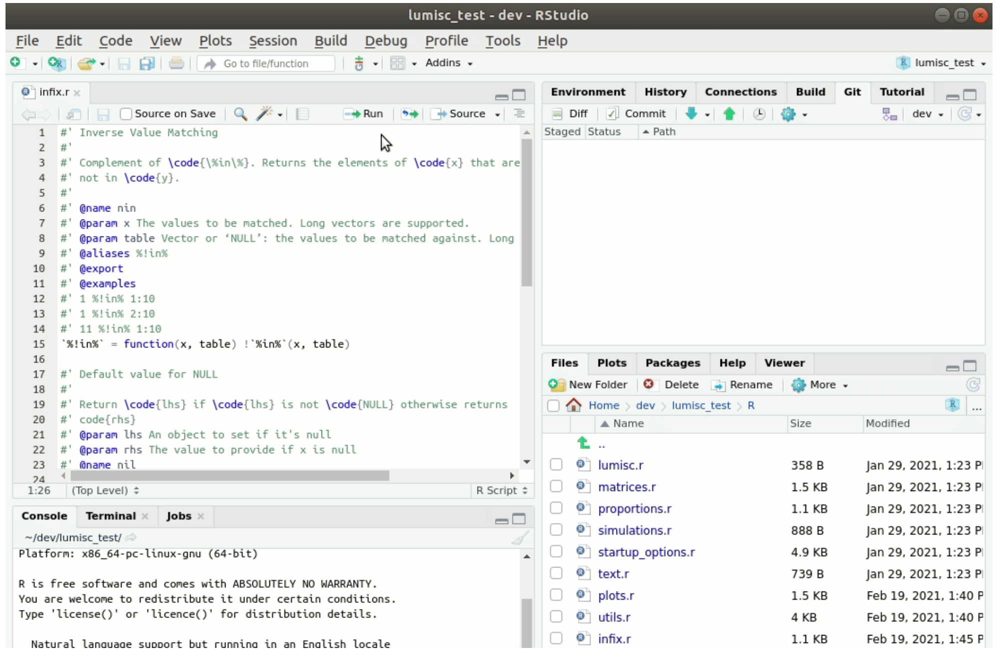
- Git makes it easy for you to test new code or change old code without having to copy/paste or save duplicate files. This is done by creating branches. A branch is an exact copy from the start of the project until the point it was created.

```
# See what files have changed
git status

# Commit our newly changed files
git branch new_branch

# To see all branches
git branch -a

# Once we are done working on our new
# branch we can delete it
git branch -d new_branch
```



```
#' Inverse Value Matching
#'
#' Complement of \code{\%in\%}. Returns the elements of \code{x} that are
#' not in \code{y}.
#'
#' @name nin
#' @param x The values to be matched. Long vectors are supported.
#' @param table Vector or ‘NULL’: the values to be matched against. Long
#' @aliases %!in%
#' @export
#' @examples
#' 1 %!in% 1:10
#' 1 %!in% 2:10
#' 11 %!in% 1:10
`%!in%` = function(x, table) !`%in%`(x, table)

#' Default value for NULL
#'
#' Return \code{lhs} if \code{lhs} is not \code{NULL} otherwise returns
#' code{rhs}
#' @param lhs An object to set if it's null
#' @param rhs The value to provide if x is null
#' @name nil
```

```
~/dev/lumisc_test/
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale
```

Name	Size	Modified
lumisc.r	358 B	Jan 29, 2021, 1:23 P
matrices.r	1.5 KB	Jan 29, 2021, 1:23 P
proportions.r	1.1 KB	Jan 29, 2021, 1:23 P
simulations.r	888 B	Jan 29, 2021, 1:23 P
startup_options.r	4.9 KB	Jan 29, 2021, 1:23 P
text.r	739 B	Jan 29, 2021, 1:23 P
plots.r	1.5 KB	Feb 19, 2021, 1:40 P
utils.r	4 KB	Feb 19, 2021, 1:40 P
infix.r	1.1 KB	Feb 19, 2021, 1:45 P

Typical Git Workflow

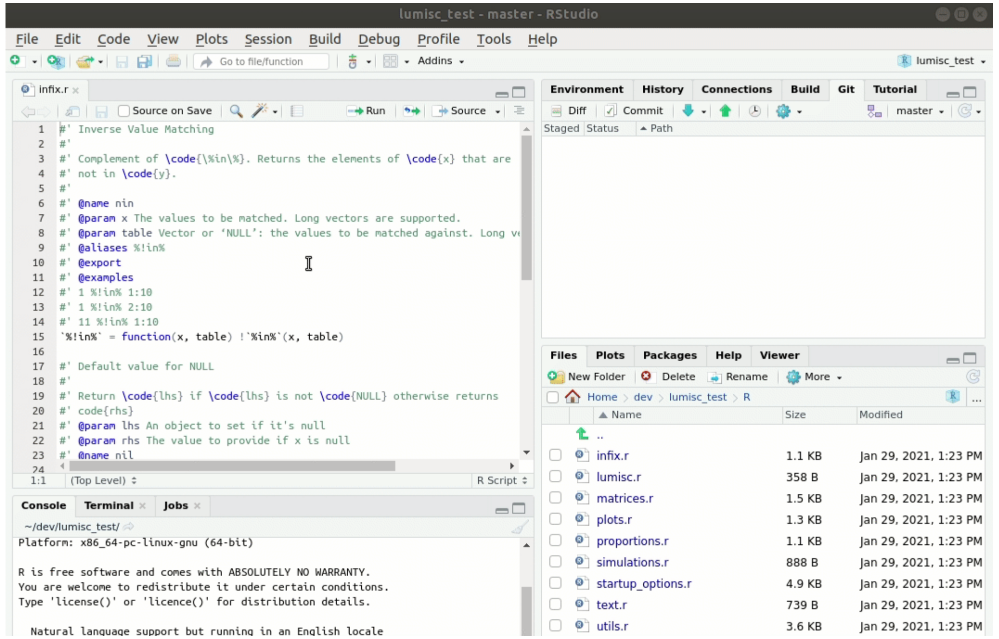
Switching Between Branches

- Once we have created between branches we can easily* switch between them.

```
# To see all branches
git branch -a

# If you have the latest version of git
git switch new_branch

# If you have an older version of git
# Older than git 2.23
git checkout new_branch
```


lumisc_test - master - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
lumisc_test
infix.r
Source on Save
Run
Source
Environment History Connections Build Git Tutorial
Diff
Commit
master
Staged Status Path
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home › dev › lumisc_test › R
1:1
(Top Level)
R Script
Console Terminal Jobs
~/dev/lumisc_test/

```
#' Inverse Value Matching
#'
#' Complement of \code{\%in\%}. Returns the elements of \code{x} that are
#' not in \code{y}.
#'
#' @name nin
#' @param x The values to be matched. Long vectors are supported.
#' @param table Vector or ‘NULL’: the values to be matched against. Long ve
#' @aliases %!in%
#' @export
#' @examples
#' 1 %!in% 1:10
#' 1 %!in% 2:10
#' 11 %!in% 1:10
`%!in%` = function(x, table) !`%in%`(x, table)

#' Default value for NULL
#'
#' Return \code{lhs} if \code{lhs} is not \code{NULL} otherwise returns
#' code{rhs}
#' @param lhs An object to set if it's null
#' @param rhs The value to provide if x is null
#' @name nil
```

```
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale
```

Name	Size	Modified
..		
infix.r	1.1 KB	Jan 29, 2021, 1:23 PM
lumisc.r	358 B	Jan 29, 2021, 1:23 PM
matrices.r	1.5 KB	Jan 29, 2021, 1:23 PM
plots.r	1.3 KB	Jan 29, 2021, 1:23 PM
proportions.r	1.1 KB	Jan 29, 2021, 1:23 PM
simulations.r	888 B	Jan 29, 2021, 1:23 PM
startup_options.r	4.9 KB	Jan 29, 2021, 1:23 PM
text.r	739 B	Jan 29, 2021, 1:23 PM
utils.r	3.6 KB	Jan 29, 2021, 1:23 PM

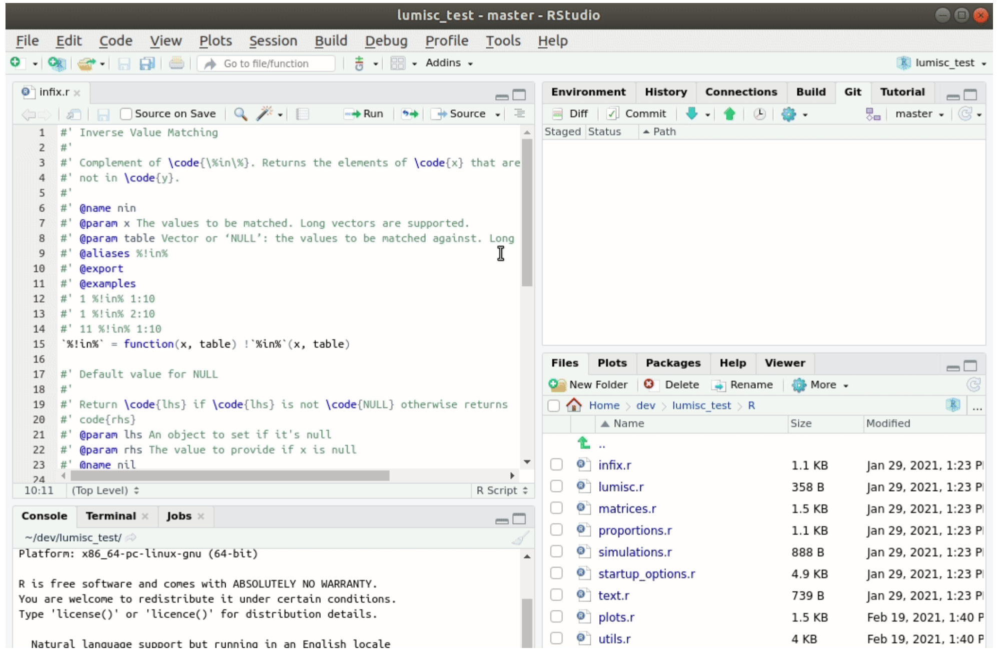
Typical Git Workflow

Viewing Your History

```
# See the entire commit history
# of your current branch
git log

# View some statistics for each commit
git log --stat

# Compress history to oneline for
# each commit for easier reading
git log --pretty=oneline
```


lumisc_test - master - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
lumisc_test
infix.r
Source on Save
Run
Source
Environment History Connections Build Git Tutorial
Diff Commit
master
Staged Status Path
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home › dev › lumisc_test › R
Console Terminal Jobs
~/dev/lumisc_test/
10:11 (Top Level) R Script

```
#' Inverse Value Matching
#'
#' Complement of \code{\%in\%}. Returns the elements of \code{x} that are
#' not in \code{y}.
#'
#' @name nin
#' @param x The values to be matched. Long vectors are supported.
#' @param table Vector or ‘NULL’: the values to be matched against. Long
#' @aliases %!in%
#' @export
#' @examples
#' 1 %!in% 1:10
#' 1 %!in% 2:10
#' 11 %!in% 1:10
`%!in%` = function(x, table) !`%in%`(x, table)

#' Default value for NULL
#'
#' Return \code{lhs} if \code{lhs} is not \code{NULL} otherwise returns
#' code{rhs}
#' @param lhs An object to set if it's null
#' @param rhs The value to provide if x is null
#' @name nil
```

```
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale
```

Name	Size	Modified
..		
infix.r	1.1 KB	Jan 29, 2021, 1:23 P
lumisc.r	358 B	Jan 29, 2021, 1:23 P
matrices.r	1.5 KB	Jan 29, 2021, 1:23 P
proportions.r	1.1 KB	Jan 29, 2021, 1:23 P
simulations.r	888 B	Jan 29, 2021, 1:23 P
startup_options.r	4.9 KB	Jan 29, 2021, 1:23 P
text.r	739 B	Jan 29, 2021, 1:23 P
plots.r	1.5 KB	Feb 19, 2021, 1:40 P
utils.r	4 KB	Feb 19, 2021, 1:40 P

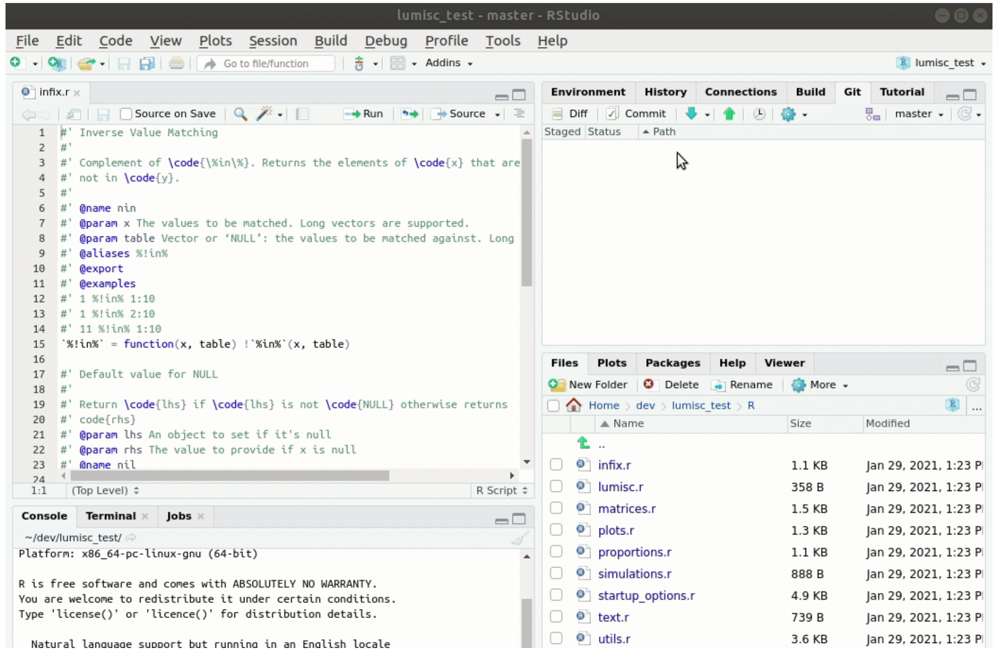
Typical GitHub/GitLab Workflow

Updating Your Local Repository

- This only applies when you are working with a git repository that is stored on some server or cloud service (like **GitHub or GitLab**). This is usually done **once per work session** before you start making changes to make sure your local version is the same as the online repository.

```
# Pull the latest changes from the remote
# branch
git pull

# Sometimes git will warn you that you
# have to pick what to do if there
# are conflicts between changed files
# in your local repository and changed files
# in the remote repository
git pull --ff-only
# Will only pull if there are no conflicts
# otherwise it will error out
```



```
#' Inverse Value Matching
#'
#' Complement of \code{\%in\%}. Returns the elements of \code{x} that are
#' not in \code{y}.
#'
#' @name nin
#' @param x The values to be matched. Long vectors are supported.
#' @param table Vector or ‘NULL’: the values to be matched against. Long
#' @aliases %!in%
#' @export
#' @examples
#' 1 %!in% 1:10
#' 1 %!in% 2:10
#' 11 %!in% 1:10
`%!in%` = function(x, table) !`%in%`(x, table)

#' Default value for NULL
#'
#' Return \code{lhs} if \code{lhs} is not \code{NULL} otherwise returns
#' code{rhs}
#' @param lhs An object to set if it's null
#' @param rhs The value to provide if x is null
#' @name nil
```

```
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale
```

Name	Size	Modified
..		
infix.r	1.1 KB	Jan 29, 2021, 1:23 P
lumisc.r	358 B	Jan 29, 2021, 1:23 P
matrices.r	1.5 KB	Jan 29, 2021, 1:23 P
plots.r	1.3 KB	Jan 29, 2021, 1:23 P
proportions.r	1.1 KB	Jan 29, 2021, 1:23 P
simulations.r	888 B	Jan 29, 2021, 1:23 P
startup_options.r	4.9 KB	Jan 29, 2021, 1:23 P
text.r	739 B	Jan 29, 2021, 1:23 P
utils.r	3.6 KB	Jan 29, 2021, 1:23 P

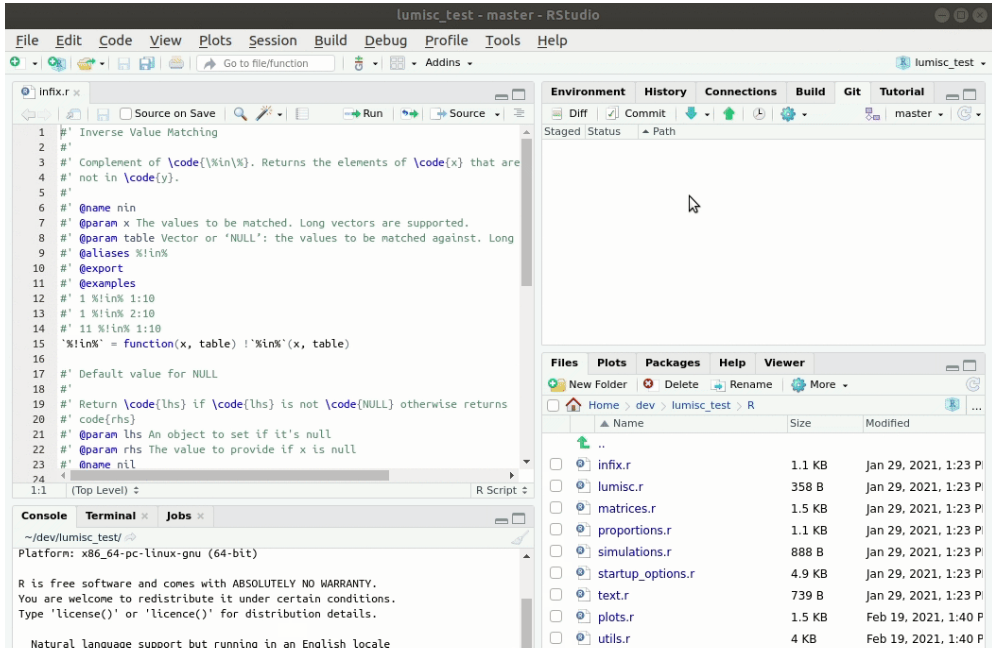
Typical GitHub/GitLab Workflow

Updating The Remote Repo (i.e GitHub)

- Again this only applies when dealing with a repo stored on a cloud server or GitHub. This is also usually only done **once per work session** at the end of your work session when you want to make sure your changes are recorded on the remote server.

```
# Push the latest changes to the remote
# repo
git push

# If there are conflicts between your
# local repo and the remote repo Git will
# error and give you a chance to
# fix the errors
git push -f
# Will overwrite the remote repo with
# your local changes and is usually
# not recommended
```


lumisc_test - master - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
lumisc_test
infix.r
Source on Save
Run
Source
Environment History Connections Build Git Tutorial
Diff Commit
master
Staged Status Path
Console Terminal Jobs
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home › dev › lumisc_test › R

```
#' Inverse Value Matching
#'
#' Complement of \code{\%in\%}. Returns the elements of \code{x} that are
#' not in \code{y}.
#'
#' @name nin
#' @param x The values to be matched. Long vectors are supported.
#' @param table Vector or ‘NULL’: the values to be matched against. Long
#' @aliases %!in%
#' @export
#' @examples
#' 1 %!in% 1:10
#' 1 %!in% 2:10
#' 11 %!in% 1:10
`%!in%` = function(x, table) !`%in%`(x, table)

#' Default value for NULL
#'
#' Return \code{lhs} if \code{lhs} is not \code{NULL} otherwise returns
#' code{rhs}
#' @param lhs An object to set if it's null
#' @param rhs The value to provide if x is null
#' @name nil
```

1:1 (Top Level) R Script

```
~/dev/lumisc_test/
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale
```

Name	Size	Modified
..		
infix.r	1.1 KB	Jan 29, 2021, 1:23 P
lumisc.r	358 B	Jan 29, 2021, 1:23 P
matrices.r	1.5 KB	Jan 29, 2021, 1:23 P
proportions.r	1.1 KB	Jan 29, 2021, 1:23 P
simulations.r	888 B	Jan 29, 2021, 1:23 P
startup_options.r	4.9 KB	Jan 29, 2021, 1:23 P
text.r	739 B	Jan 29, 2021, 1:23 P
plots.r	1.5 KB	Feb 19, 2021, 1:40 P
utils.r	4 KB	Feb 19, 2021, 1:40 P

Advanced Git

HERE BE DRAGONS

THESE SLIDES ARE ONLY FOR REFERENCE

Advanced Git Tricks

That you should almost never have to use

- Ask me, Luke, about any complicated or tricky Git Stuff.
- You can go back to anypoint in your git history.
    - Find the **commit hash** using git log.

```
git checkout "2b1d1b15f46ad71b59"
```

- It can be useful to export/archive all the code in a given git repository.

```
git archive -o filename.zip branch_name
```

End

Thank You for Your Time

Glossary

Version Control: any system that allows you to understand the history of the file and how it has progressed

Git: a version control program which allows you to annotate the changes you make to create an easily traversable system history

Commit: an annotated “snapshot” of the differences made to the system at a given point in time

Local: refers to the computer you’re working on this very minute

Remote: refers to an online location

Repository (repo): a special folder configured with Git superpowers containing all the files pertaining to your project/system

Glossary

Github: takes your local commit history and hosts it remotely so that you can access it from any computer

Pushing: the action of taking local Git commits (and whatever work these encompass) and putting them online on Github

Pulling: the action of taking online Github commits and bringing them into your local machine

Master (branch): the “trunk” of the commit history “tree”; contains all approved content/code

Feature branch: an isolated location, based off of master, where you can write a new piece of work safely before reincorporating said changes back to master

Pull Request: a Github tool that allows users to easily see the changes (the difference or “diff”) that a feature branch is proposing as well as discuss any tweaks that said branch might require before it is merged into master

Merging: the action of taking the commits from a feature branch and adding them to the top of master’s history

Checking out: the action of moving from one branch to another

Extra Resources for Learning

- Git Basics
- Git Visualized
- Interactive Git Tutorial