

Intro2R-Git

Version control in R using Git and Github

Song, Xiao Ping

xp.song@u.nus.edu

Course materials: <https://github.com/xp-song/Intro2R-Git>

updated 2021-09-28



Intro2R-Git

1. Navigate to course webpage for instructions and background information
<https://github.com/xp-song/Intro2R-Git>
2. [Download](#) workshop materials
(green button on webpage)

Outline

Why use version control?

Git with RStudio Projects

Initial set up

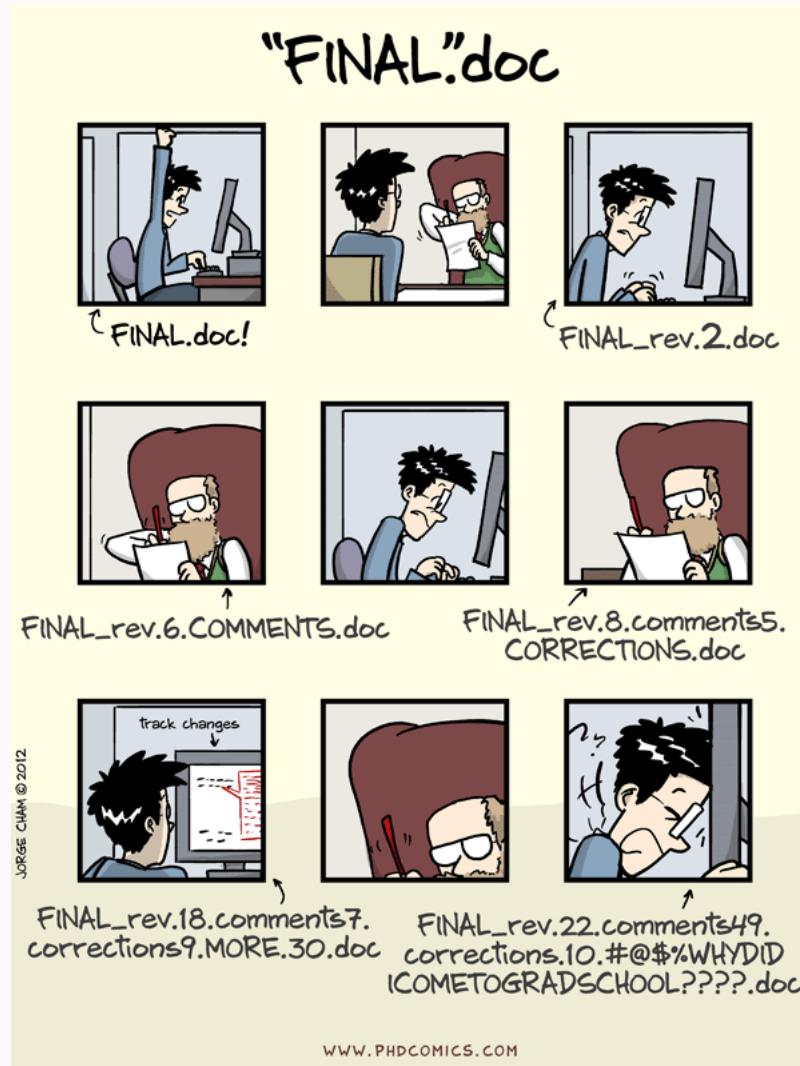
Git it going

Git basics

Git collaborating

Useful resources

Why use version control?

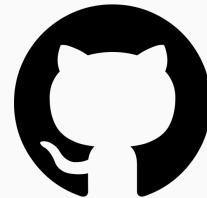


Why use version control?



Git

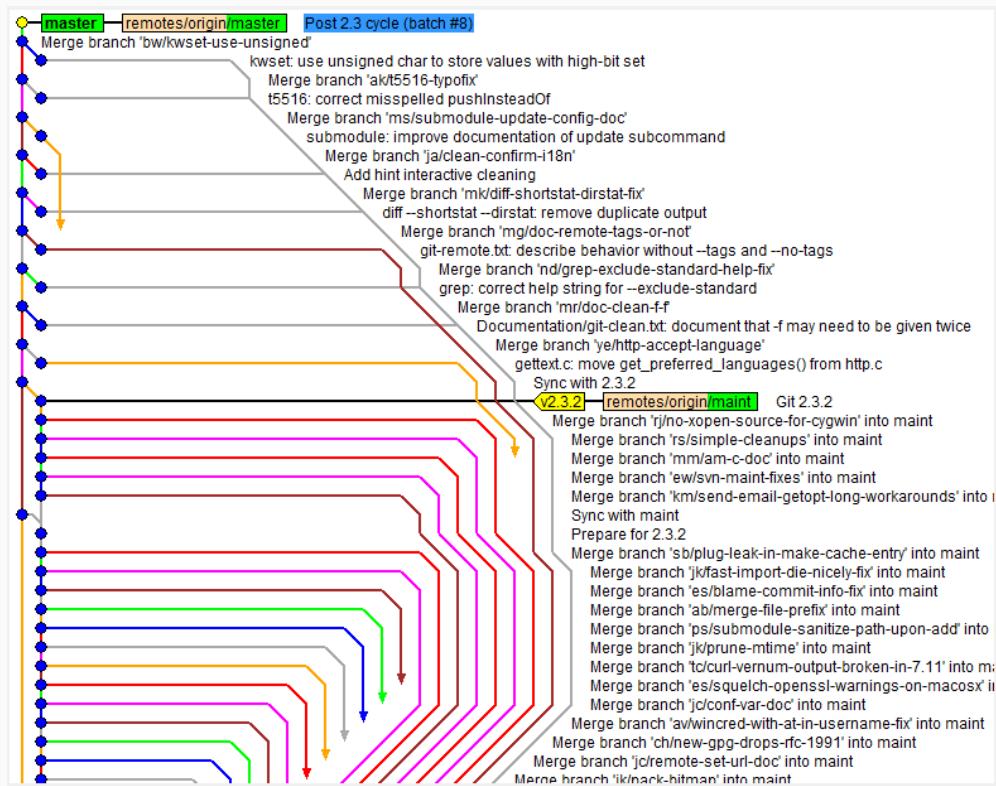
- Version control system
- *Track* changes to your code
- *Visualise* changes with GUI clients
(e.g. RStudio, Github Desktop, etc.)
- *Share* changes & collaborate
(e.g. Github, Gitlab, Bitbucket)
- Changes are combined automatically
- Systematic way to de-conflict changes



Git + Github

- Widely used in software development
- Most popular for developing R packages

A bit about Git



Git hell (tugberkugurlu.com)

Outline

Why use version control?

Git with RStudio Projects

Initial set up

Git it going

Git basics

Git collaborating

Useful resources

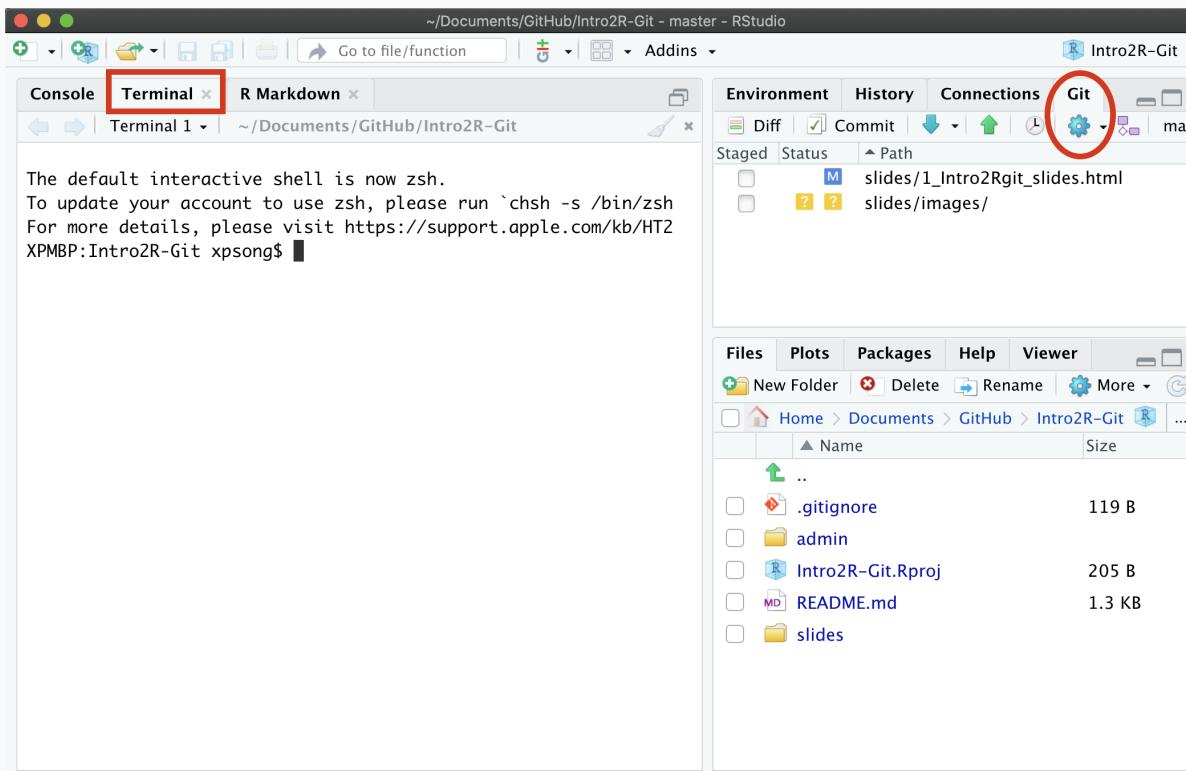
Git with RStudio Projects

- RStudio Projects help organise your work into separate 'R sessions'!
- Each project has it's own workspace a.k.a. 'working directory' (separate configuration files, history, etc.)
- The location of the '*.RProj*' file defines the 'working directory' and the '**Git repository**'

⭐ Best Practice

- Use *relative* paths in your script, based on *.RProj* file location
- Keep all project items in the working directory

Git with RStudio Projects



- If needed, you can access the Shell via *Tools > Shell* (or the '*Terminal*' tab)
- Commands will be executed in the working directory (location of the '*.RProj*' file)

Important Shell commands: `pwd` to print working directory; `cd <name>` : change directory, or `cd ..` to move up the directory hierarchy; `ls` : list files in the current directory.

Outline

Why use version control?

Git with RStudio Projects

Initial set up

Git it going

Git basics

Git collaborating

Useful resources

Initial set up

Install Git

- OS X: <http://git-scm.com/download/mac>
- Windows: <http://git-scm.com/download/win>
- Debian/Ubuntu: sudo apt-get install git-core
- Other Linux distros: <http://git-scm.com/download/linux>

Initial set up

Install Git

Set-up Git

In the *Shell*, configure your name and email for Git:

```
git config --global user.name "YOUR FULL NAME"
```

```
git config --global user.email "YOUR EMAIL ADDRESS"
```

All commits (i.e. tracked changes) you make will be labelled with this information.

You can check if you're set up correctly by running:

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

Initial set up

Install Git

Create an account at <https://github.com>

Set-up Git

Github

Initial set up

Install Git

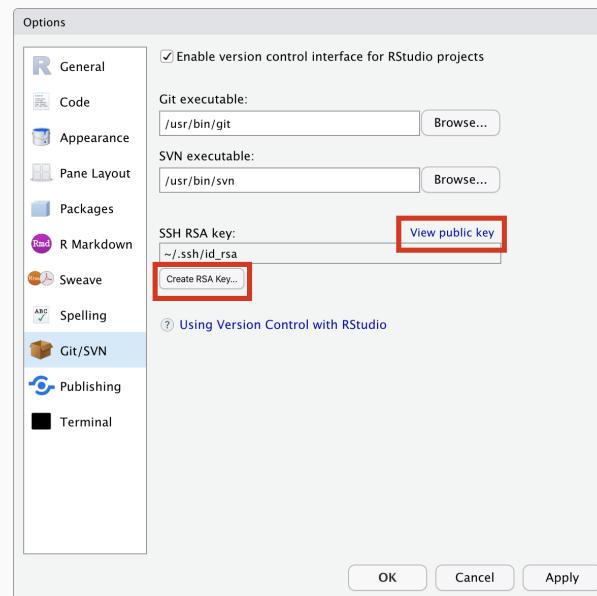
Set-up Git

Github

Security

Use an **SSH key-pair** to reduce the need to key in your password

- The *public* key encrypts data to be read by those with *private* key
- In R, check if it is already present: `file.exists("~/ssh/id_rsa.pub")`
- Go to *RStudio Preferences* to create/view the public key:



Initial set up

Install Git

Add SSH key to Github at <https://github.com/settings/ssh>

Set-up Git

Github

Security

SSH keys

New SSH key

There are no SSH keys associated with your account.

Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH Problems](#).

GPG keys

New GPG key

There are no GPG keys associated with your account.

Learn how to [generate a GPG key and add it to your account](#).

Note: If you've turned on 2FA on Github, [create a Personal Access Token \(PAT\)](#) under the 'Developer settings' of your 'Account settings'. Use the generated PAT as the password within RStudio, instead of your regular password.

Outline

Why use version control?

Git with RStudio Projects

Initial set up

Git it going

Git basics

Git collaborating

Useful resources

Intro2R-Git

Unzip [downloaded](#) workshop materials and open the RStudio Project file
Intro2R-Git.Rproj

Initialise Git repo

The screenshot shows the RStudio interface with the following details:

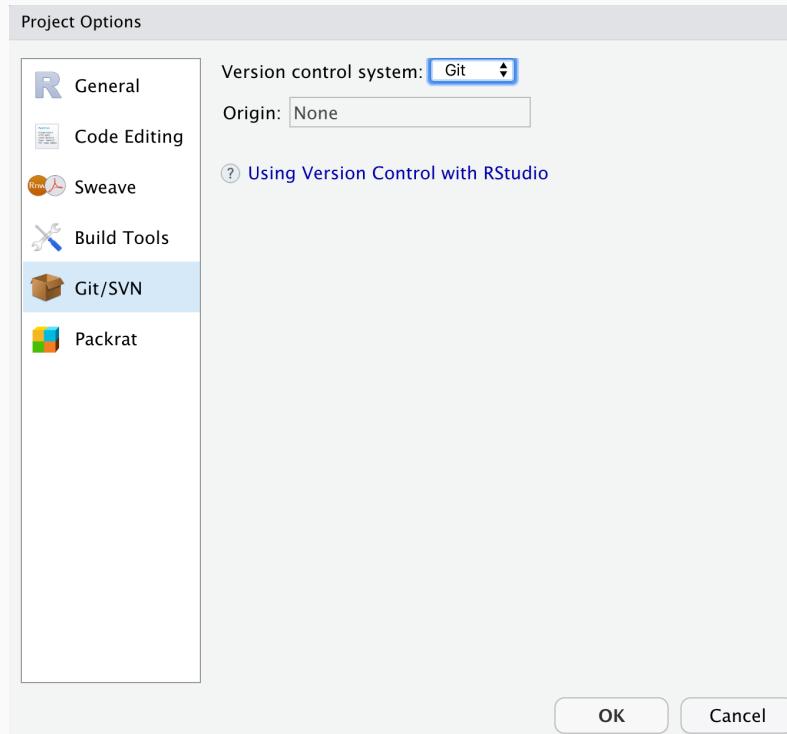
- Header Bar:** Shows the path `~/Desktop/Intro2R-Git-master - RStudio` and the project name `Intro2R-Git-master`.
- Console Tab:** Displays the R startup message for version 4.0.0 (2020-04-24) "Arbor Day".
- Global Environment:** Shows that the environment is currently empty.
- Files View:** Lists the contents of the `Intro2R-Git-master` directory:
 - .. (Parent directory)
 - .gitignore (119 B)
 - Intro2R-Git.Rproj (205 B)
 - README.md (1.7 KB)
 - slides (Folder)

A red circle highlights the "List" button in the Global Environment tab, which typically displays a list of objects in the current environment.

Initialise Git repo

Local Git: Enable version control with RStudio Project

Go to *Tools > Project Options > Git/SVN* (you will be prompted to restart R).

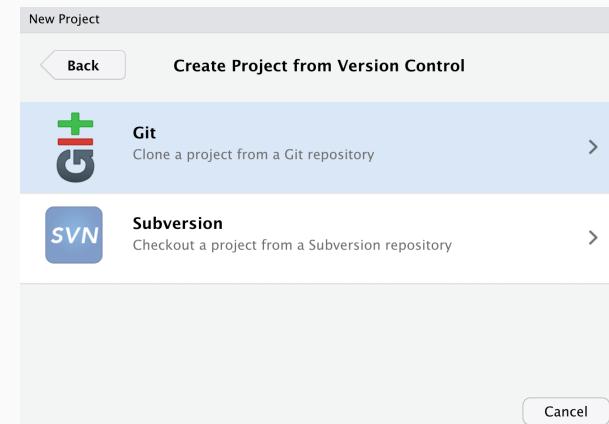
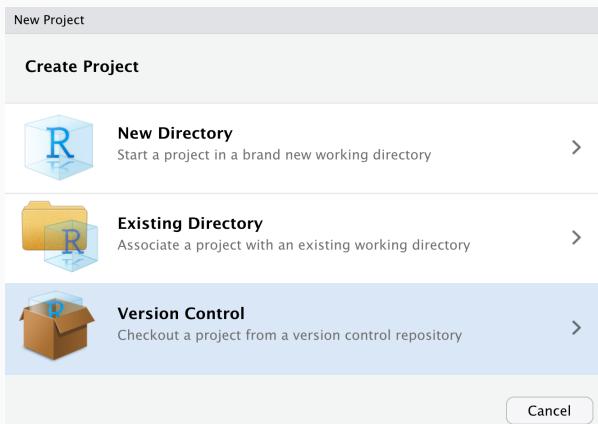


Alternative method: run `git init` in shell

Initialise Git repo

Local Git + Github: Clone remote repo directly from Github using RStudio

Go to *File > New Project*



Initialise Git repo

Local Git + Github: Clone remote repo directly from Github using RStudio

Go to *File > New Project*

The screenshot shows a GitHub repository page for 'xp-song/Intro2R-Git'. The 'Clone with HTTPS' URL, <https://github.com/xp-song/Intro2R-Git.git>, is highlighted with a red box. Other options like 'Use SSH' and 'Download ZIP' are also visible.

The screenshot shows the 'Clone Git Repository' dialog in RStudio. The 'Repository URL:' field contains the value <https://github.com/xp-song/Intro2R-Git.git>. A red arrow points to this field. Below it, the 'Project directory name:' field is set to 'Intro2R-Git'. The 'Create project as subdirectory of:' field has the value '/Documents/GitHub'. At the bottom, the 'Open in new session' checkbox is checked and circled with a red oval. There are 'Create Project' and 'Cancel' buttons at the bottom right.

Outline

Why use version control?

Git with RStudio Projects

Initial set up

Git it going

Git basics

Git collaborating

Useful resources

Git basics

Create a new R Notebook at **File > New File > R Notebook**, and save it as **test.Rmd**

The screenshot shows the RStudio interface with the following details:

- Editor Area:** Displays the content of the `test.Rmd` file. It includes code chunks, rendered output, and some explanatory text.
- Git Status Window:** Located on the right side of the interface. It shows the current state of files in the repository:
 - Staged:** A list of files with status indicators (A, ? (yellow), M (blue), D (red)) and their paths:
 - .gitignore
 - Intro2R-Git.Rproj
 - slides/
 - test.Rmd** (highlighted with a blue selection bar)
 - test.nb.html
 - Files:** A list of tracked files with their names, sizes, and modification times.
- Console:** Located at the bottom of the interface.

git status

- Yellow : New untracked file
- Green: File added
- Red: File deleted
- Blue: File modified

Git basics

The screenshot shows a Git commit interface with the following elements:

- Top Bar:** Changes, History, master, Stage, Revert, Ignore, Pull, Push.
- Staged Changes List:** .gitignore (Untracked), Intro2R-Git.Rproj (Untracked), slides/ (Untracked), test.Rmd (Modified), test.nb.html (Deleted).
- Commit Message:** A large text area for entering the commit message, with an "Amend previous commit" checkbox and a "Commit" button.
- Show Options:** Staged (radio selected), Unstaged, Context, 5 line, Ignore Whitespace, Stage All (checked), Discard All.
- Diff View:** Shows the differences between two versions of a file. Lines 11-15 show the addition of a new chunk of code. Lines 16-17 show a note about saving the notebook. Lines 18-19 show added text. Buttons for "Stage chunk" and "Discard chunk" are visible above the diff.

git diff

- Background colors show sections that are added/removed
- Shows old/new line numbers
- Greyed-out sections give you context

Git basics

Commit changes



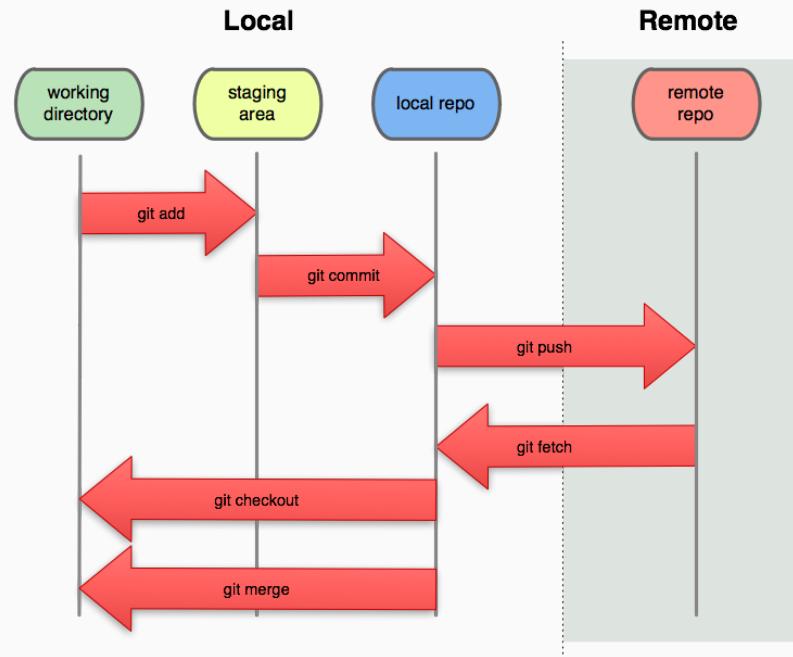
`git commit` has five components:

1. Unique ID aka SHA (secure hash algorithm)
2. Author
3. Commit message
4. Changeset that describes files that were added/modified/deleted
5. One parent (i.e. previous) commit, unless merging commits from multiple parents or if this is the initial commit

Git basics

Commit changes

Changes must be '*staged*', in preparation for a '*commit*'.



Git basic operations (kevintshoemaker.github.io)

Git basics

Commit changes

The screenshot shows a Git commit interface with the following details:

- Changes Tab:** Shows the current state of files:
 - Staged:** .gitignore, Intro2R-Git.Rproj, slides/, test.Rmd (selected), test.nb.html
 - Unstaged:** None
- Commit message:** Minor text changes to Rmd file, to show example of making commits
 - Add summary of cars dataset
 - Remove para on saving notebook
 - Add text at end of doc
- Commit Buttons:** Pull, Push, Commit, Amend previous commit
- Show Options:** Staged (radio selected), Unstaged, Context, 5 line, Ignore Whitespace, Stage All, Discard All
- Diff Viewer:** Shows the difference between the last 6 lines and the first 18 lines of the output:

```
@@ -6,13 +6,18 @@ output: html_notebook
6 6 This is an [R Markdown](http://rmarkdown.rstudio.com) Notebook. When you execute code within the
notebook, the results appear beneath the code.
7 7
8 8 Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor
inside it and pressing *Cmd+Shift+Enter*.
9 9
10 10 ````{r}
11 11 summary(cars)
12 12 plot(cars)
13 13 ````
```

Line 16 is highlighted in red with a note: "When you save the notebook, an HTML file containing the code and output will be saved alongside it (Click the *Preview* button or press *Cmd+Shift+K* to preview the HTML file)."

Line 17 is highlighted in light green with a note: "The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike *Knit*, *Preview* does not run any R code chunks. Instead, the output of the chunk when it was last run in the editor is displayed."

Line 19 is highlighted in light green with a note: "Here is some added text...".

1. Save changes
2. Select files or sections of a file to stage
3. Write commit message (<50 char header; list details)

Git basics

Commit changes



Git basics

Historic changes

The screenshot shows a Git commit history interface. At the top, there are tabs for 'Changes' (selected), 'History', and 'master'. A dropdown menu shows '(all commits)'. There are search and pull buttons. Below the tabs is a list of commits:

Author	Date	Commit SHA
Xiao Ping SONG <xp.song@u...	2018-10-17	aea7d073
Dan Rex <dan_richards@hotn...	2018-10-17	9b5d2149
Xiao Ping SONG <xp.song@u...	2018-10-17	9084ec1f
drexrichards <43661641+dre...	2018-10-17	770c3adf
Dan Rex <dan_richards@hotn...	2018-10-17	02e413f2
Dan Rex <dan_richards@hotn...	2018-10-17	da6867dc
xp-song <xp.song@u.nus.edi...	2018-10-11	f23c434c
Xiao Ping SONG <xp.song@u...	2018-10-10	57d3880b
Xiao Ping SONG <xp.song@u...	2018-10-10	ee6fa1a9
xp-song <xp.song@u.nus.edi...	2018-10-10	cd014c72
Dan Rex <dan_richards@hotn...	2018-10-10	962e6554
xp-song <xp.song@u.nus.edi...	2018-10-10	aad5bb12

Below the commit list is a summary: 'Commits 201-268 of 268'. There are navigation icons for back, forward, and search.

A detailed view of a commit is shown at the bottom:

SHA	937e9148
Author	Hao Ran <hrlai.ecology@gmail.com>
Date	2019-10-08 16:56
Subject	remove large model file
Parent	0bcb6cef

The commit message is:

```
clean_data/mixed_model_list.Rdata
@@ -1,3 +0,0 @@
1 version https://git-lfs.github.com/spec/v1
2 oid sha256:26713cf73d3c56cc4faa49e492b44841b020a22fb61884357d925c1c514ee7e3
3 size 224704600
```

To revert to a previous commit:

1. Copy the SHA
2. Run `git checkout <SHA> <filename>` in the Shell

Git basics

Ignore files specified in `.gitignore`

The screenshot shows the RStudio interface with the following details:

- Editor Area:** Displays an R Markdown file named "test.Rmd". The code includes chunks for `summary(cars)` and `plot(cars)`, and a note about adding new chunks via the toolbar or keyboard.
- Git Tab:** Shows the "Staged" status for the file ".gitignore".
- File Browser:** Shows the directory structure of the project "Intro2R-Git-master". A red box highlights the ".gitignore" file in the list of files.
- Bottom Navigation:** Includes tabs for "Files", "Plots", "Packages", "Help", "Viewer", and "Console".

Name	Size	Last Modified
<code>.gitignore</code>	119 B	Mar 20 2018
<code>.Rhistory</code>	0 B	Mar 20 2018
<code>Intro2R-Git.Rproj</code>	205 B	Mar 20 2018
<code>README.md</code>	1.7 KB	Mar 20 2018
<code>slides</code>		
<code>test.Rmd</code>	735 B	Mar 20 2018
<code>test.nb.html</code>	654.8 KB	Mar 20 2018

Outline

Why use version control?

Git with RStudio Projects

Initial set up

Git it going

Git basics

Git collaborating

Useful resources

Git collaborating

Link Github

1. [Create a new repo on Github](#)

Git collaborating

Link Github

2. Create linked local repo

Option 1: Create new RStudio project by cloning the online repo to your computer (as shown previously)

Option 2: If you've already been working on a local repo, '*push*' (i.e. publish) content to the new online repo. Run in Shell:

```
git remote add origin git@github.com:YOUR USERNAME/REPO NAME.git  
git push -u origin master
```

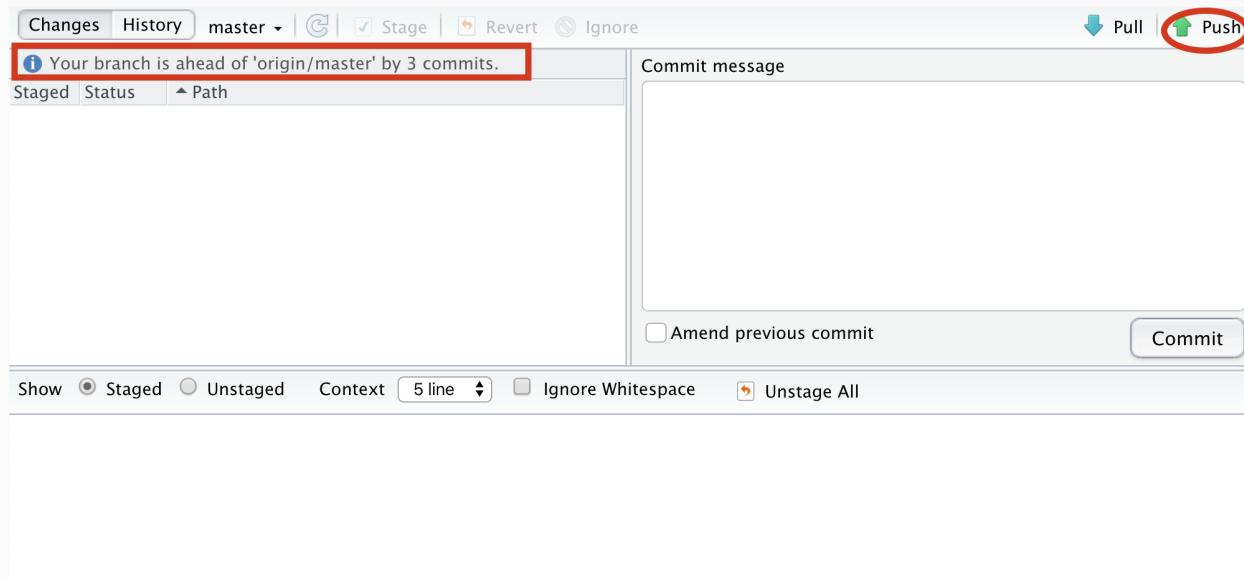
Git collaborating

Link Github

Push

Push code when you're ready for collaborators to view changes.

Each push usually includes multiple commits.



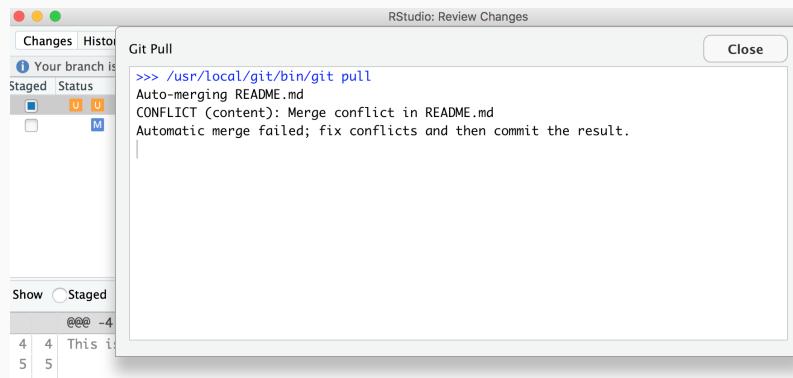
🌟 **Best Practice:** Always 'Pull' before you 'Push', in case collaborators have made changes online!

Git collaborating

Link Github

Push

Conflicts 😱



Set style to diff3

This shows local/remote changes *with the original code*. Run in the Shell:

```
git config --global merge.conflictstyle diff3
```

If you're in the middle of a merge conflict, change the setting before re-trying the merge:

```
git merge --abort
```

```
git config --global merge.conflictstyle diff3
```

```
git pull
```

Git collaborating

Link Github

De-conflict regions with conflict marker

Push

Conflicts 😱

```
<<<<< HEAD  
# This code section shows my local changes...  
|||||| merged common ancestors  
=====  
# This code section shows remote changes...  
>>>>> remote
```

The middle section shows *original* code before the split

Delete all conflict markers before staging/commit/pushing the new de-conflicted file!

If the whole thing is one hot mess, abort the merge with `git merge --abort` & try again `git pull`.

Git collaborating

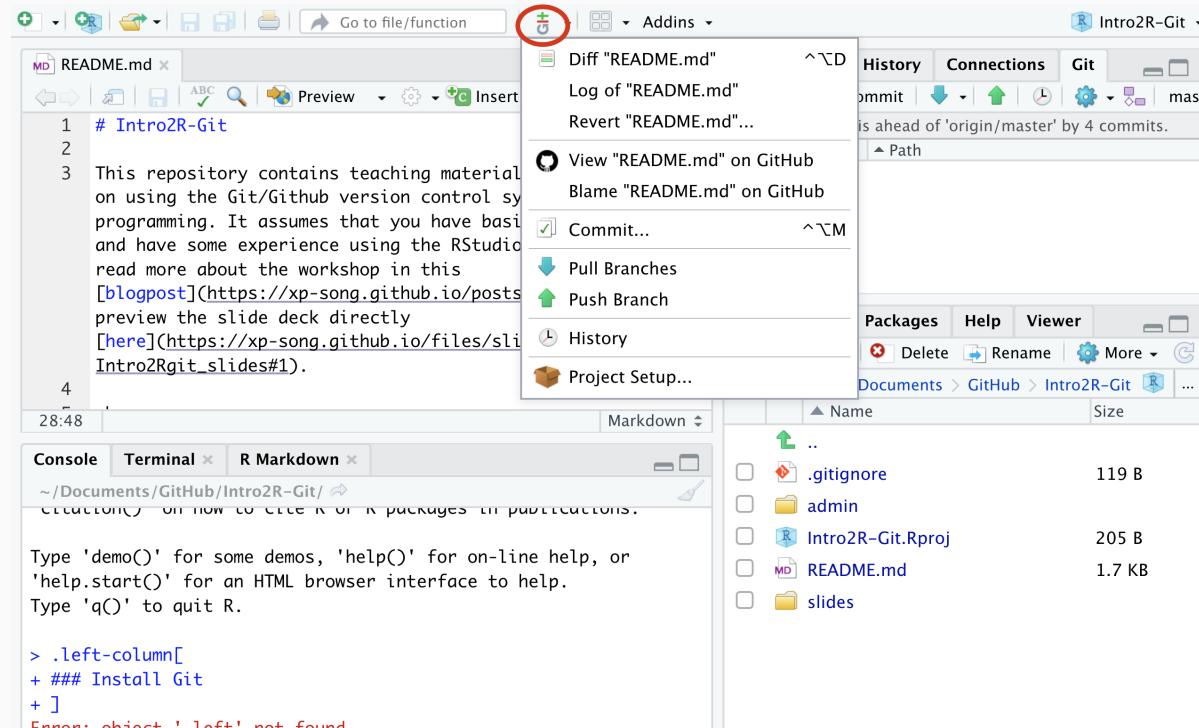
Link Github

Push

Conflicts

Investigate

Track the history of a *specific* file using the Git dropdown menu



'Blame' shows the last change made to each line of code, who made the change, and the commit the change belongs to.

Git collaborating

Link Github

Push

Conflicts

Investigate

Discuss & track issues on Github

The screenshot shows the GitHub interface for the repository `xp-song / Intro2R-Git`. The top navigation bar includes links for Pull requests, Issues, Marketplace, and Explore. On the right, there are buttons for Unwatch (1), Star (0), Fork (0), and Settings. The main navigation bar below the repository name has tabs for Code, Issues (0, highlighted with a red arrow), Pull requests (0), Actions, Projects (0), Wiki, Security (0), Insights, and Settings. A search bar at the bottom left contains the filters "is:issue is:open". The central area displays a message: "Welcome to Issues! Issues are used to track todos, bugs, feature requests, and more. As issues are created, they'll appear here in a searchable and filterable list. To get started, you should [create an issue](#)." At the bottom, a note states: "Note: Automatically link to an issue by typing `#<issue number>` in your commit message. You can also close an issue by typing `Closes #<issue number>`."

Note: Automatically link to an issue by typing `#<issue number>` in your commit message. You can also close an issue by typing
`Closes #<issue number>`.

Git collaborating

Link Github

Push

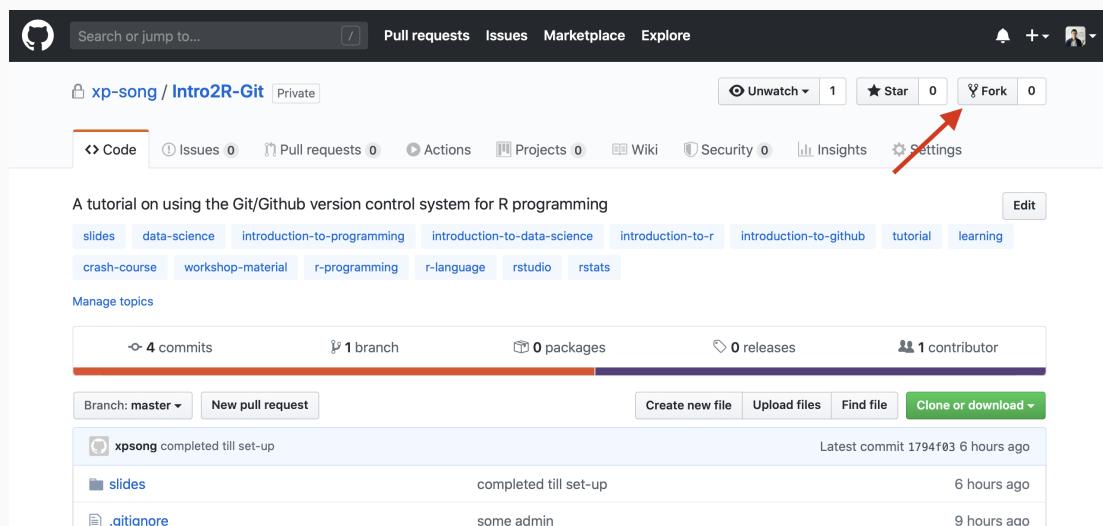
Conflicts

Investigate

Branches

What if you do not own a repo?

'Fork' (i.e. make a copy of) that repo onto your own Github account, then *clone* it to your local computer (as shown previously)



A screenshot of a GitHub repository page for 'xp-song / Intro2R-Git'. The page shows basic repository statistics: 4 commits, 1 branch, 0 packages, 0 releases, and 1 contributor. A red arrow points to the 'Fork' button in the top right corner of the header bar.

A *forked* repo is a **static** copy, which you'll have to manually update/sync up against the *original* repo (which may have undergone changes).

Git collaborating

Link Github

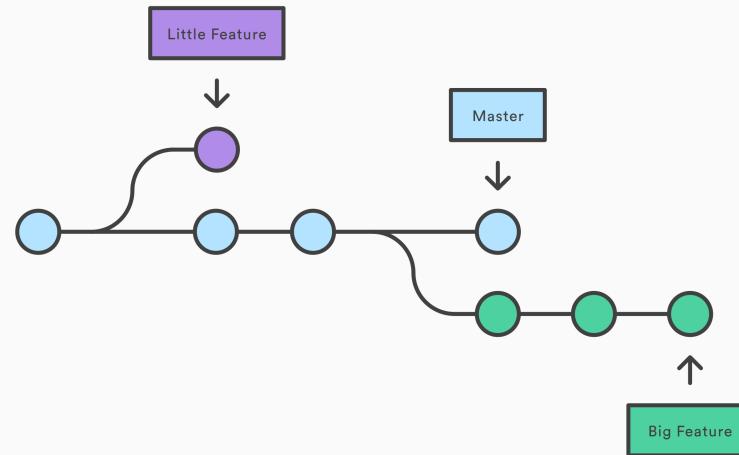
Branching in Git

Push

Conflicts

Investigate

Branches



Example showing git branches (atlassian.com)

Git collaborating

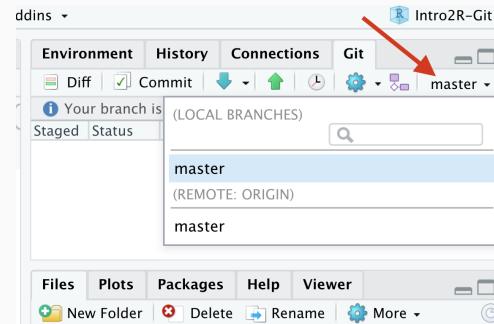
Link Github

Push

Conflicts

Investigate

Branches



Sync up with the master branch, by running in the Shell:

```
git merge master
```

To merge a branch back to the master:

```
git checkout master
```

```
git merge <branch-name>
```

```
git branch -d <branch-name>
```

Git collaborating

Link Github

Push

Conflicts

Investigate

Branches

If using a **forked** repo (you're not the owner), there are extra steps:

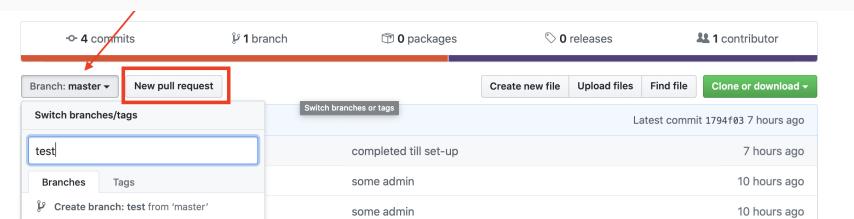
- Sync *forked* repo (on Github) with *original* repo:¹

```
git remote add upstream git@github.com:<original-name>/<repo>.git
```

```
git fetch upstream
```

```
git merge upstream/master
```

- Create a **pull request** to propose/discuss changes¹



¹ If you always submit pull requests in branches, run `git branch -u upstream/master` to create branch on your **local** repo, then sync:

```
git checkout master
```

```
git pull
```

Check for remote changes if they occur while you work on your branch:

```
git checkout <my-branch>
```

```
git merge master
```

Git collaborating

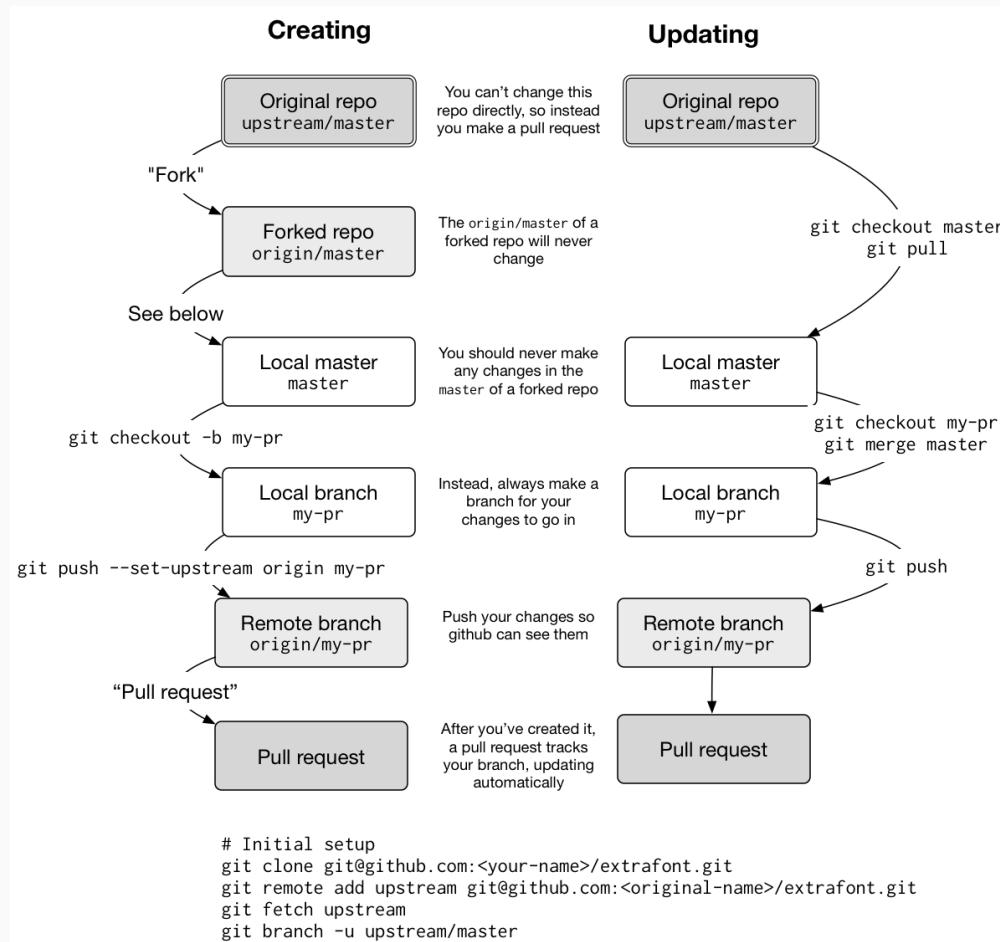
Link Github

Push

Conflicts

Investigate

Branches



Overview of pull requests (<http://r-pkgs.had.co.nz/git.html>)

Outline

Why use version control?

Git with RStudio Projects

Initial set up

Git it going

Git basics

Git collaborating

Useful resources

Useful Resources

[**One-page guide**](#) by Hadley Wickham

[**Full e-book**](#) by Jenny Bryan

[**Infographic on fixing common problems**](#)