

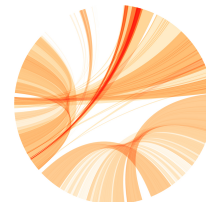
Introduction to Git & GitHub

February 2018

Stefan Wyder
stefan.wyder@uzh.ch



**Universität
Zürich** ^{UZH}



**URPP
Evolution
in Action**

Git is hard to learn (and teach)

- Git was not build for our exact usage (supports many workflows)
- Git commands are sometimes not intuitive
- The advantages are long-term
- Learning/using Git is sometimes frustrating

"Git has a real knack for making me feel stupid!"
Torsten Seemann

The problem

"FINAL".doc



FINAL.doc!



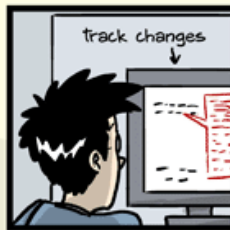
FINAL_rev.2.doc



FINAL_rev.6.COMMENTS.doc



FINAL_rev.8.comments5.
CORRECTIONS.doc



FINAL_rev.18.comments7.
corrections9.MORE.30.doc



FINAL_rev.22.comments49.
corrections.10. #@\$%WHYDID
ICOMETOGRADSCHOOL?????.doc

JORGE CHAM © 2012

Problem cont.

- You want to improve a script that works. After 2 hours of rewriting the new script still doesn't work. Unfortunately you overwrote the original version that worked and can't go back.
- You biked into the office only to realize that the USB key with last night's work is still on the kitchen table.
- Which version of the script/data did I use 3 years ago when I did the analysis?

You use Git to take snapshots of all the files in a folder (including subfolders).

This folder is called a **repository** or **repo**.

When you want to take a snapshot of a project (1 file or many files), you create a **commit**



Boxplot.R



Boxplot-2.R



Boxplot-Elena-feedback.R



Boxplot-Final.R



Boxplot-Final2.R

By saving copies



`Boxplot.R`

By making commits



`Boxplot.R`

By saving copies



Boxplot.R



Boxplot-2.R

By making commits



Boxplot.R



Boxplot.R



By saving copies



Boxplot.R



Boxplot-2.R



Boxplot-Elena-feedback.R

commit

By making commits



Boxplot.R



Boxplot.R



Boxplot.R

By saving copies



Boxplot.R



Boxplot-2.R



Boxplot-Elena-feedback.R



Boxplot-Final.R

By making commits



Boxplot.R



Boxplot.R



Boxplot.R



Boxplot.R



By saving copies



Boxplot.R



Boxplot-2.R



Boxplot-Elena-feedback.R



Boxplot-Final.R



Boxplot-Final2.R

By making commits



Boxplot.R



Boxplot.R



Boxplot.R



Boxplot.R



Boxplot.R



By saving copies



Boxplot.R



Boxplot-2.R



Boxplot-Elena-feedback.R



Boxplot-Final.R



Boxplot-Final2.R

By making commits



Boxplot.R



Boxplot.R



Boxplot.R



Boxplot.R



Boxplot.R

When you **commit** a file or files,
some information is along with the
changes to the file

1. Who
2. When
3. commit message

You can add more information about the changes you've made in a **commit message**

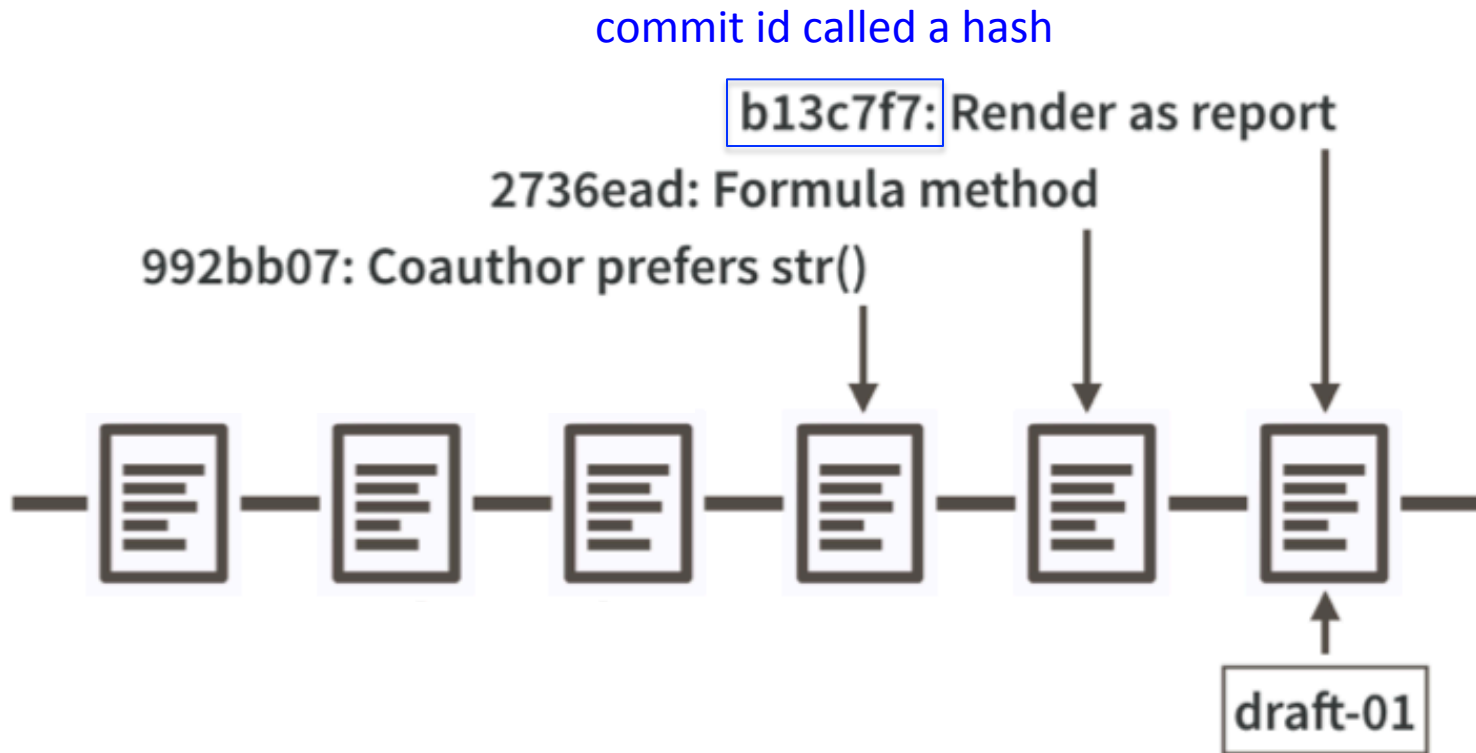
Stefan Wyder
2:25pm January 30th 2018

New colorblind-safe colors

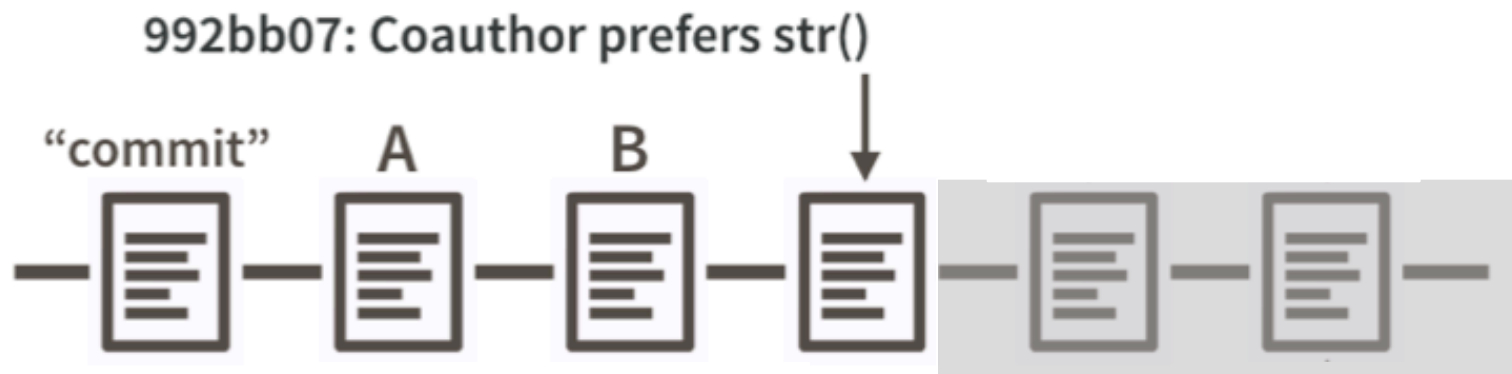
More detailed explanatory text, if necessary. Explain the problem that this commit is solving

explains what and why
~~how~~ (the diff explains)

Git stores the whole history of your project

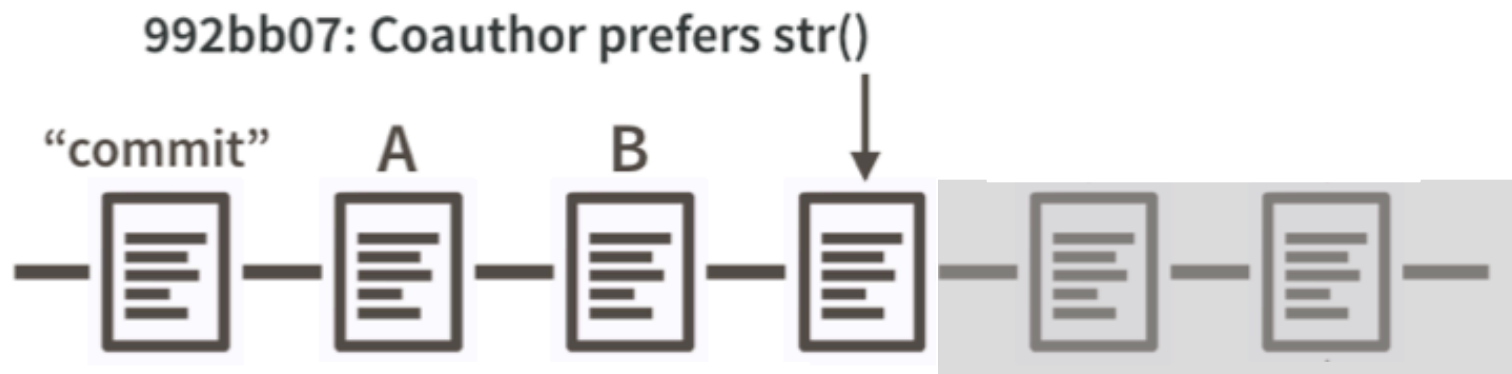


I can tell Git what commit I want to check out using the commit hash



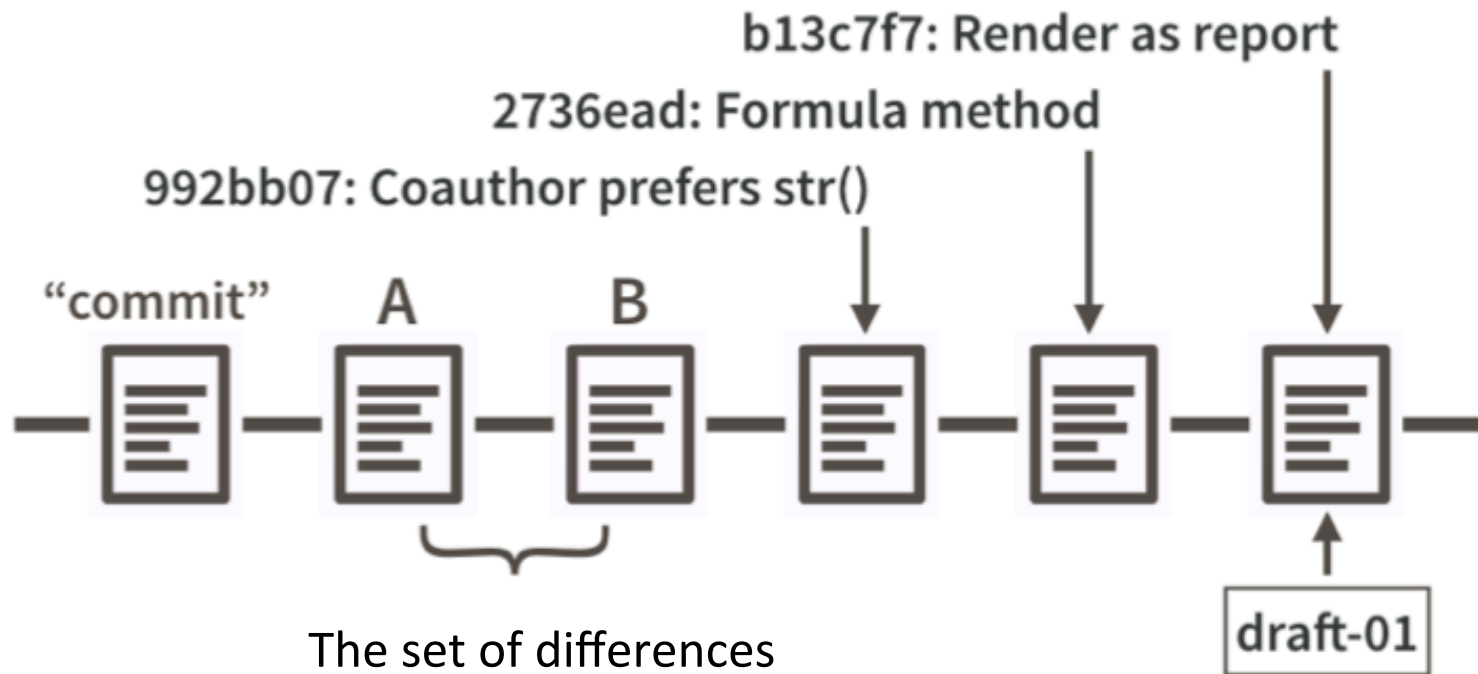
My other commits still exist, but when I look in my repo, it's as they never happened

I can tell Git what commit I want to check out using the commit hash



My other commits still exist, but when I look in my repo, it's as they never happened

The power of Git comes from the **diff**



The diff

Showing 1 changed file with 7 additions and 7 deletions.

Unified

Split

View



14 Day2/3_1-functions-and-modules.ipynb

@@ -115,19 +115,19 @@

```
115     "cell_type": "markdown",
116     "metadata": {},
117     "source": [
118 -     "### Using Modules\n",
119     "\n",
120 -     "One of the great things about Python is the free availability of a _huge_
number of modules that can be imported into your code and used. Modules are
developed with the aim of solving some particular problem or providing particular,
often domain-specific, capabilities.\n",
121     "\n",
122 -     "Like functions, which are usable parts of a program, packages (also known as
libraries) are reusable programs with several modules.\n",
123     "\n",
124 -     "In order to import a module, it must first be installed and available on your
system. We will cover this briefly later in the course. \n",
```

```
115     "cell_type": "markdown",
116     "metadata": {},
117     "source": [
118 +     "### Libraries and Modules\n",
119     "\n",
120 +     "One of the great things about Python is the free availability of a _huge_
number of libraries (also called package) that can be imported into your code and
(re)used. \n",
121     "\n",
122 +     "Modules contain functions for use by other programs and are developed with
the aim of solving some particular problem or providing particular, often domain-
specific, capabilities. A library is a collection of modules, but the terms are
often used interchangeably, especially since many libraries only consist of a
single module (so don't worry if you mix them). \n",
123     "\n",
124 +     "In order to import a library, it must available on your system or should be
installed. \n",
```

Version control works for **any file format** (code, pdfs, docs, xls, jpg, etc.)

... but works best for plain text-based files (txt, Markdown, Python, R, Matlab, any other code)

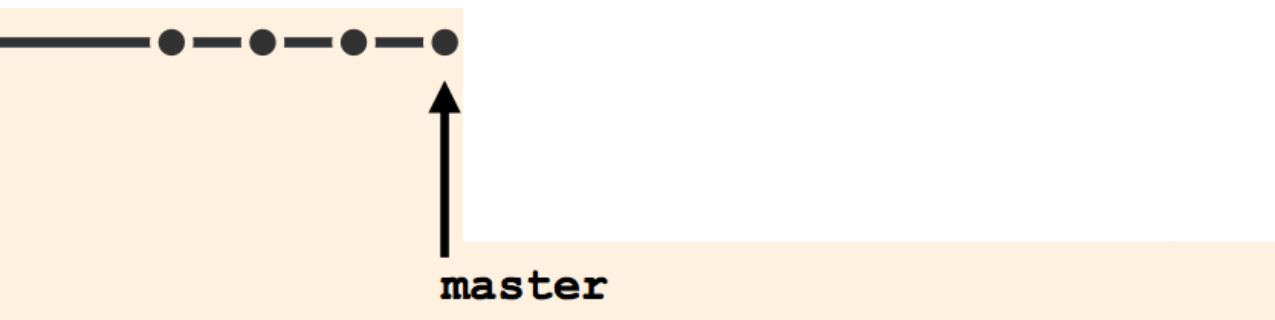
Git helps you experiment

So far, everything has been very linear and ordered.

But sometimes you want to make easily discardable experiments

The way you do this in Git is with **branches**

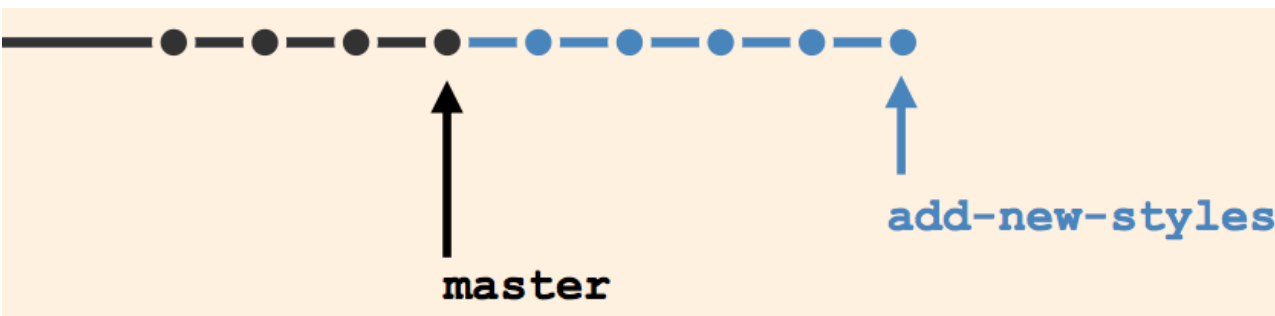
The default branch name in Git is
master



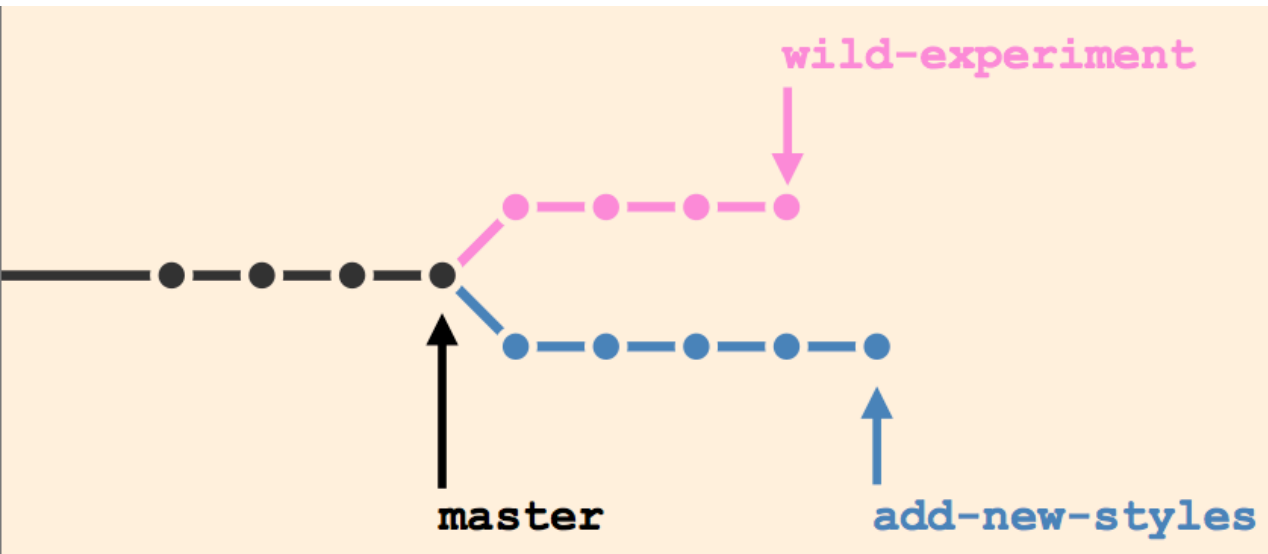
You can add your own branches too



and do lots of work on them

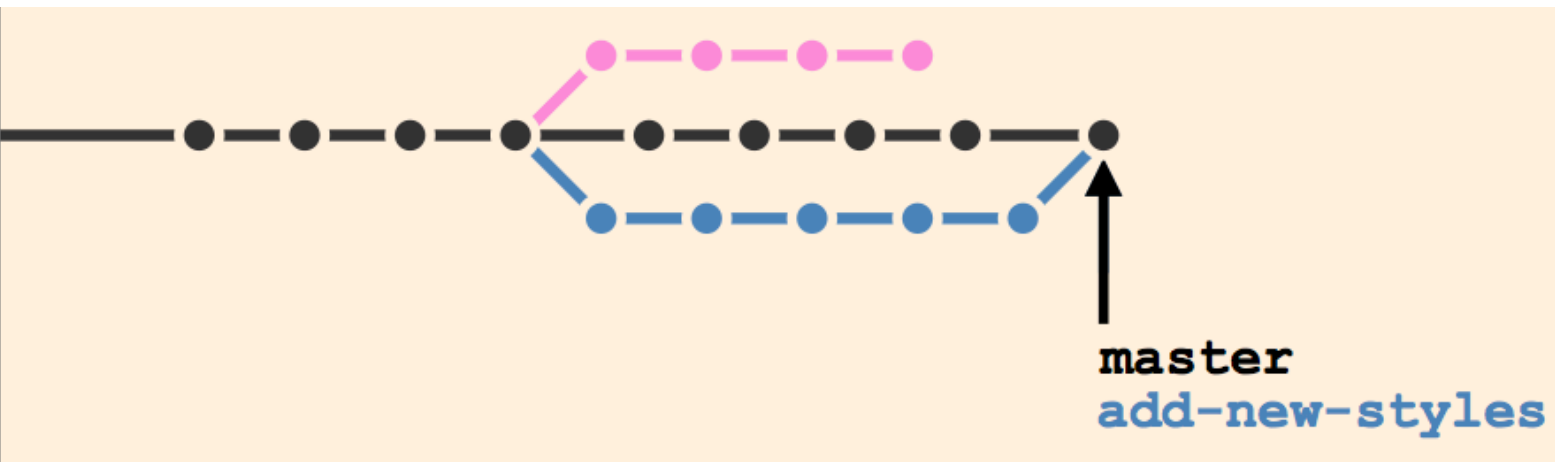


Branches are useful for trying out stuff



The master branch is often considered special (latest stable version) and other branches contain work in progress

Once you're happy with some work, you need a way to get it back to master

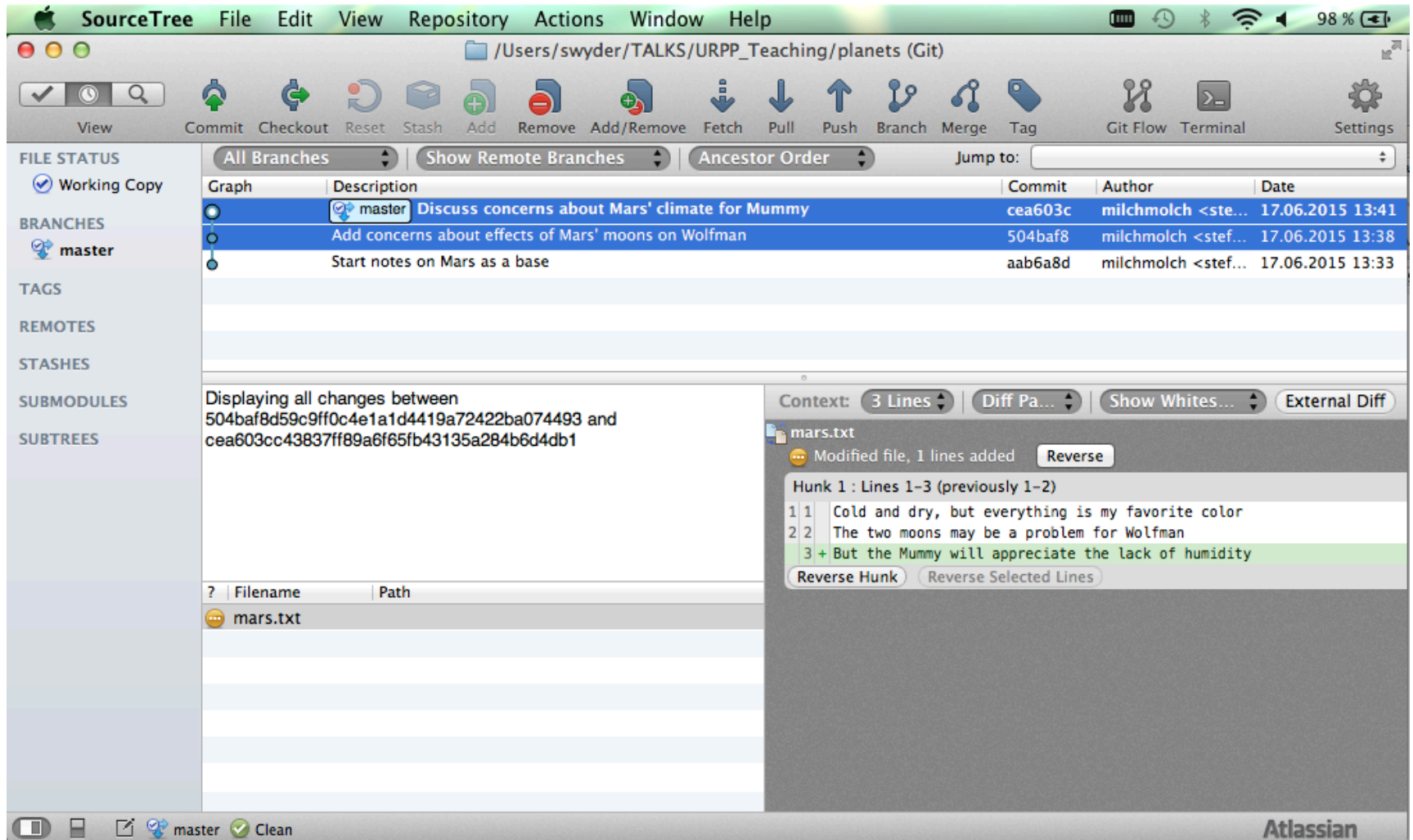


To get changes from one branch into another, you merge them



1. Git lets you tell the story of your project ([commit](#): take snapshots of files)
2. Git lets you time travel ([check out](#))
3. Git helps you experiment ([branch](#))
4. Git helps you backup your project
5. Git helps you collaborate across multiple computers/people ([clone push pull merge](#))

Try out graphical front-end



e.g. SourceTree or GitKraken

Basic git commands in Rstudio

The image shows two overlapping windows from the RStudio application. The 'Options' window on the left is set to the 'General' tab, with the 'Git/SVN' section selected. It shows settings for enabling version control, specifying git and svn executables, and an SSH RSA key. The 'Review Changes' window on the right displays a list of commits for the 'master' branch. The selected commit (SHA 1ef1b330) is titled 'More git writing.' and shows a diff for the file 'git.rmd'.

Options - General

- ☒ Enable version control interface for RStudio projects
- Git executable:
- SVN executable:
- SSH RSA Key: [View public key](#)
-
- [Using Version Control with RStudio](#)

Review Changes

Changes History master (all commits) Search Refresh Pull

SHA	Author	Date	Subject	Parent
1ef1b330	hadley <h.wickham@gmail.com>	2014-12-04 16:56	More git writing.	cbc29305
cbc29305	hadley <h.wickham@gmail.com>	2014-12-03	Add hint from @jennybc	
acc83457	hadley <h.wickham@gmail.com>	2014-12-03	More git rewriting	
59935c4d	hadley <h.wickham@gmail.com>	2014-12-03	Add parts to book	
f7e5fcbc	hadley <h.wickham@gmail.com>	2014-12-03	Fix minor wrapping probs	

Commits 1-100 of 524

git.rmd [View file @ 1ef1b330](#)

```
@@ -159,27 +159,27 @@ The background colour tells you whether the text has
been added (green) or remov

159 159
160 160 In the shell, use `git status` to show an overview of changes and `git diff`
to see detailed differences.
161 161
162 ## Add and commit files {#git-stage-commit}
162 ## Record changes {#git-commit}
163 163
164 The fundamental unit of work in git is a __commit__. A commit takes snapshot
of your code at a specified point in time. If you make a mistake, you can
```

<https://support.rstudio.com/hc/en-us/articles/200532077-Version-Control-with-Git-and-SVN>
<http://r-pkgs.had.co.nz/git.html>

Git terms

repository	your project folder
commit	a snapshot of your repo
hash	an id for a commit
checkout	time travel to a specific commit
branch	a movable label that points to a commit
merge	combining two branches
remote	a computer with the repository on it
clone	get the repository from the remote for the first time
push	send commits to a remote
pull	get commits from a remote

Sources

- <http://cdn.rawgit.com/luispedro/talk-git-intro/master/slides.html>
- <https://speakerdeck.com/alicebartlett/git-for-humans>
- <https://doi.org/10.7287/peerj.preprints.3159v2>

Advantages of Git/GitHub vs Google docs