

INTRODUCTION TO

[Kasia Kędzierska](#)

9/11/2020, WHG, University of Oxford

WHAT IS VERSION CONTROL?

version control (aka revision control or source control) is the management of changes to documents, computer programs, large web sites, and other collections of information. [...]

Each revision is associated with a **timestamp** and the **person making the change**.

Revisions can be compared, **restored**, and with some types of files, merged.

Source: [Wikipedia](#), accessed: 12/12/2019

WHAT IS VERSION CONTROL?

Tracks **changes** and allows for comparisons

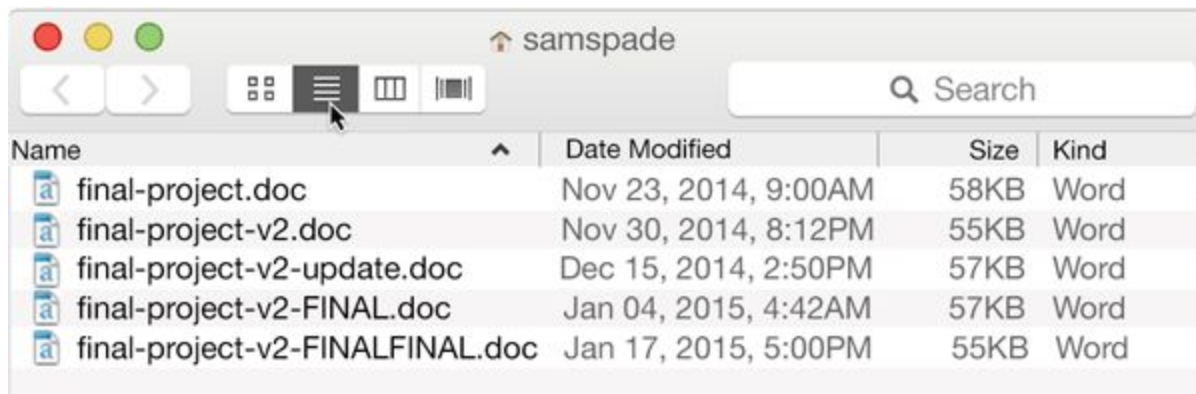
Keeps informative **details** about introduced changes

when? who? what?

Allows to **revert** to chosen version


WHY USE VERSION CONTROL?

TO CHANGE THIS...






Source: [How to name things](#) by (amazing) [Jenny Bryan](#)


INTO THIS

 Commits on Mar 9, 2018


Updated README



 **aakrosh** committed on 9 Mar 2018


 **e4c5770** 

 Commits on Mar 7, 2018


Monte carlo (#11) ...



 **kzkedzierska** committed on 7 Mar 2018


Verified  **02bc6e0** 

 Commits on Feb 22, 2018


fixed handling empty predicted read-outs



 **kzedzierska** committed on 22 Feb 2018


 **ba330b9** 

 Commits on Feb 21, 2018


introduced all three conditions and did some changes to the output (#9) ...



 **kzkedzierska** committed on 21 Feb 2018

Verified  **5bef7f1** 

 Commits on Feb 19, 2018

adding option for logging intermediate simulation results (#8) ...

 **kzkedzierska** committed on 19 Feb 2018

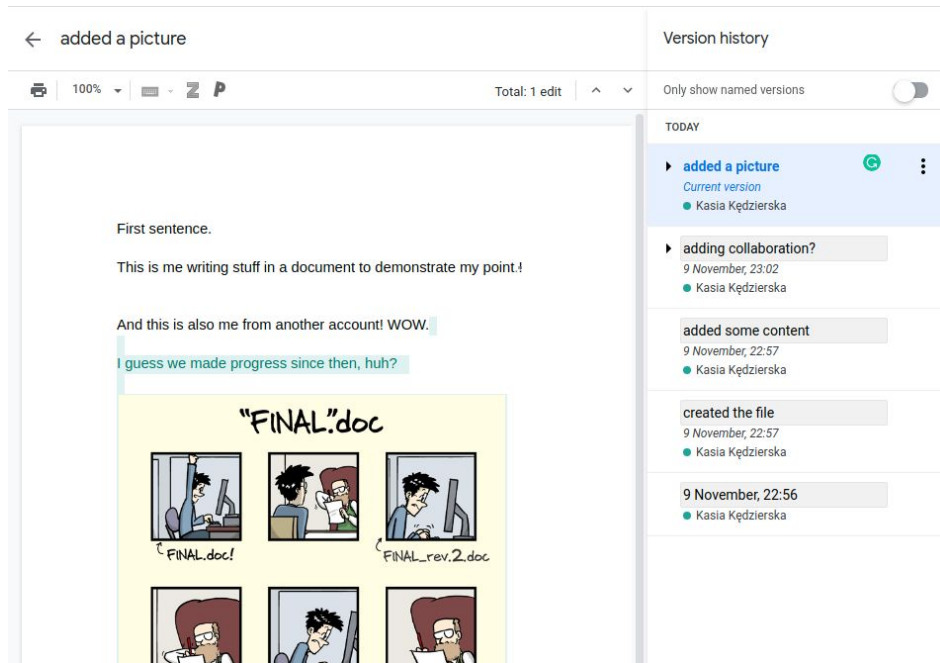
Verified  **8eeecb1** 

ALSO, YOU MIGHT BE ALREADY USING VERSION CONTROL

In GDocs one can **track changes**, collaborate add comments and suggestions.

One can also **name** versions, **revert** to particular edits and make a copy.

I hear some of that is available in MS Word as well, but being a Linux user I can't know for sure... :D



The screenshot displays the Google Docs interface for a document titled "added a picture". The top right corner shows the "Version history" tab, which is currently active. Below the tab, there is a toggle switch for "Only show named versions" and a "TODAY" section. The version history list on the right includes the following entries:

- added a picture** (Current version) by Kasia Kędzierska
- adding collaboration?** by Kasia Kędzierska (9 November, 23:02)
- added some content** by Kasia Kędzierska (9 November, 22:57)
- created the file** by Kasia Kędzierska (9 November, 22:57)
- 9 November, 22:56** by Kasia Kędzierska

The main document content on the left includes the following text:

First sentence.

This is me writing stuff in a document to demonstrate my point!

And this is also me from another account! WOW.

I guess we made progress since then, huh?

Below the text is a comic strip titled "FINAL.doc" showing a sequence of events. The comic is divided into two rows of three panels each. The top row shows a man at a computer, a woman at a computer, and a man at a computer. The bottom row shows a woman at a computer, a man at a computer, and a woman at a computer. The comic is labeled "FINAL.doc" and "FINAL_rev.2.doc".

WHAT'S GIT THEN?

WHAT'S GIT THEN?

NAME

git - the stupid content tracker



git

--distributed-even-if-your-workflow-isnt

GIT - DISTRIBUTED VERSION CONTROL

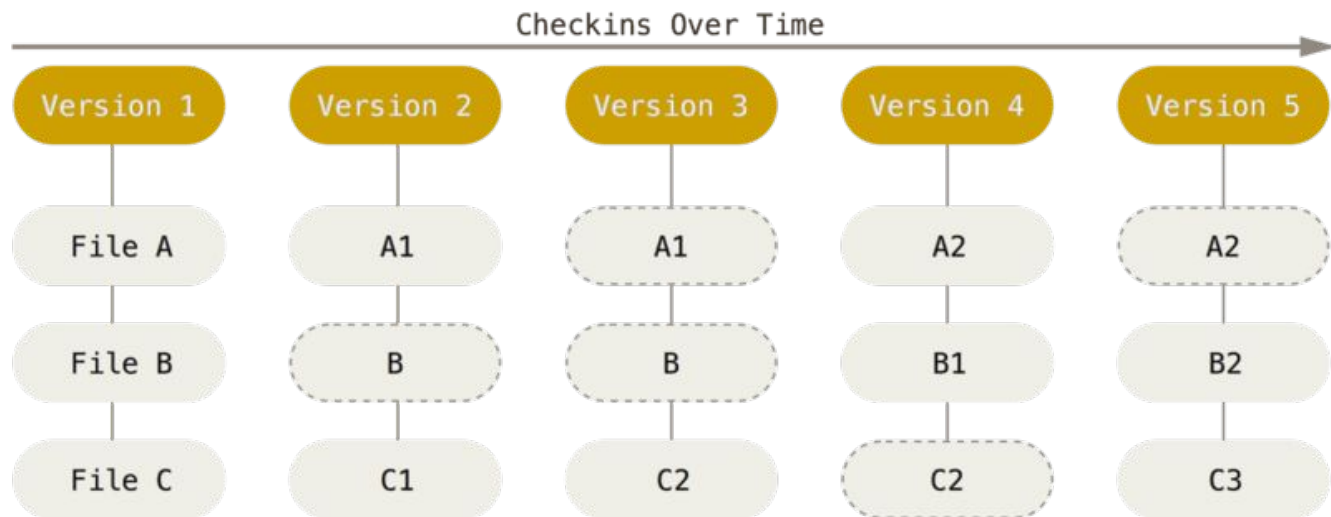


Git was created by Linus Torvalds in 2005 for development of the Linux kernel.

When asked why he called the new software, "git," British slang meaning "a rotten person," he said. "I'm an egotistical bastard, so I name all my projects after myself. First Linux, now git."

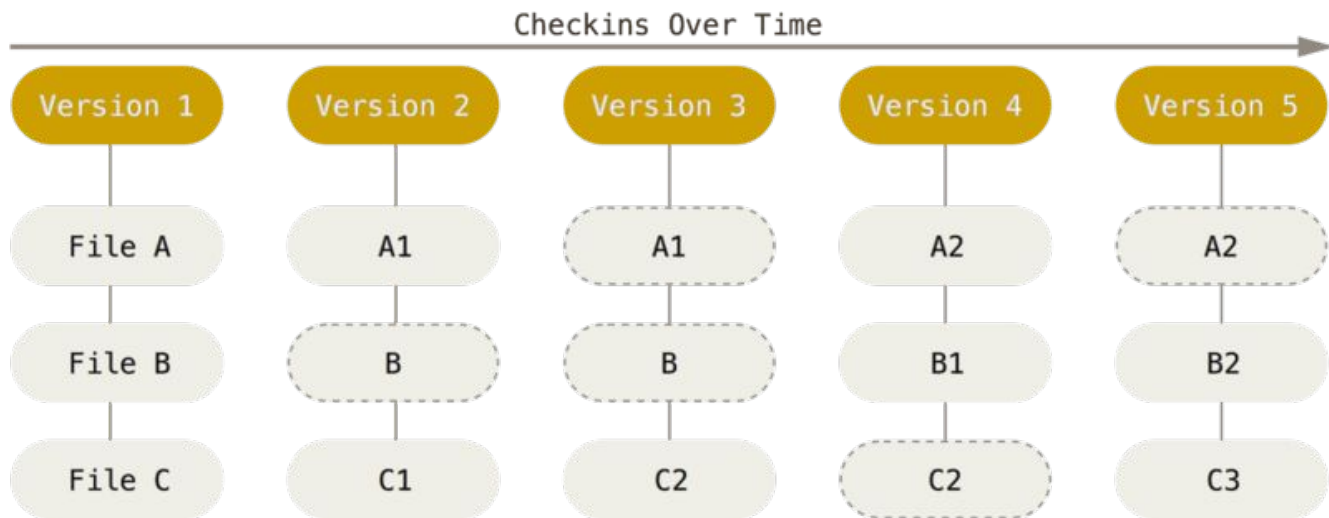
GIT - TAKING SNAPSHOTS OF THE PROJECT

At each time we **commit** changes git takes a snapshot of the repository.



GIT - TAKING SNAPSHOTS OF THE PROJECT

To save the space it doesn't copy unchanged files, instead creates links to those.



3 GIT STATES

All the files tracked by git can exist in one of the 3 states:

- **modified** - a file was created, changes were made
- **staged** - git was made aware of the file
- **committed** - changes were saved to current commit

IS GIT HARD?



WELL, YES AND NO

Yes, there are more than **180 git commands**.

But, for everyday use you can get around with as little as around **10 of them**.

TYPICAL GIT WORKFLOW

initialize repository

add changes

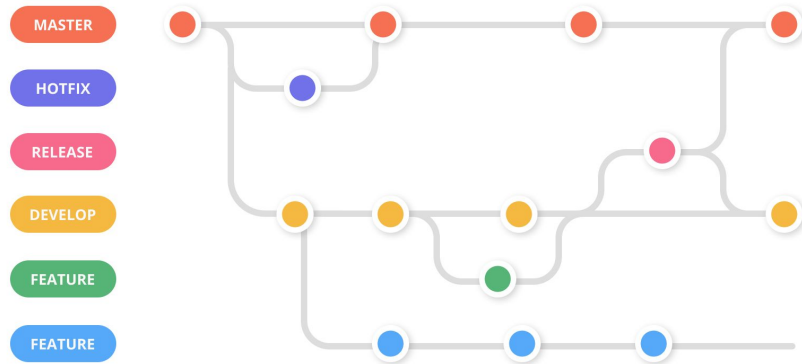
commit them to *memory*

push them to remote

branch out and test ideas

merge the side changes

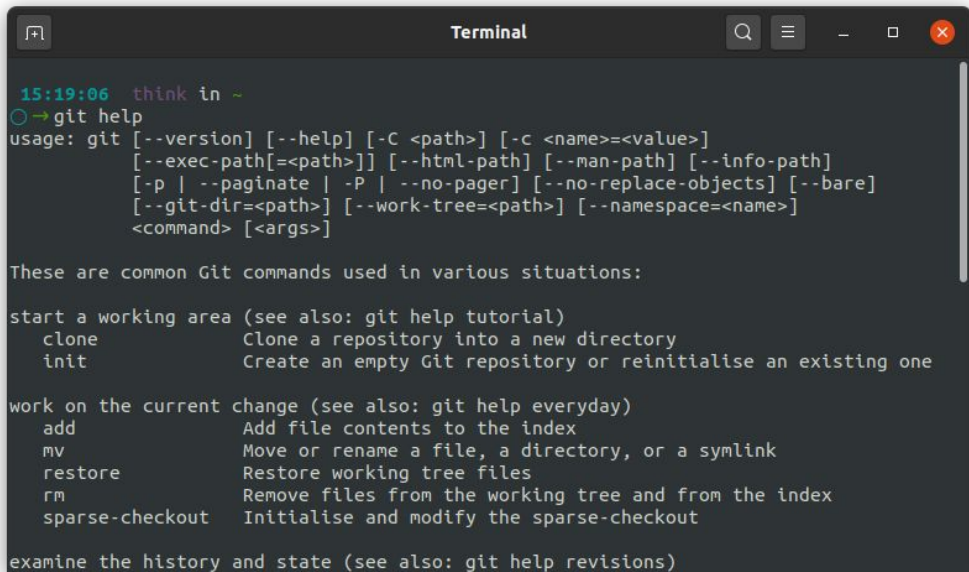
push the changes



LET'S GET STARTED THEN

GIT HELP

It's always good to know we can ask for help, `git help` is there for us

A terminal window titled "Terminal" with standard window controls (search, menu, zoom, close). The prompt is "15:19:06 think in ~". The command "git help" has been executed. The output shows the usage of the 'git' command with various options like --version, --help, -C, -c, --exec-path, --html-path, --man-path, --info-path, --paginate, --no-pager, --no-replace-objects, --bare, --git-dir, --work-tree, --namespace, and <command> <args>. Below the usage, it lists common Git commands categorized into three groups: starting a working area (clone, init), working on the current change (add, mv, restore, rm, sparse-checkout), and examining the history and state (not explicitly listed as a category but implied by the text).

```
15:19:06 think in ~
➜ git help
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
      [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
      [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
      [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
      <command> [<args>]

These are common Git commands used in various situations:

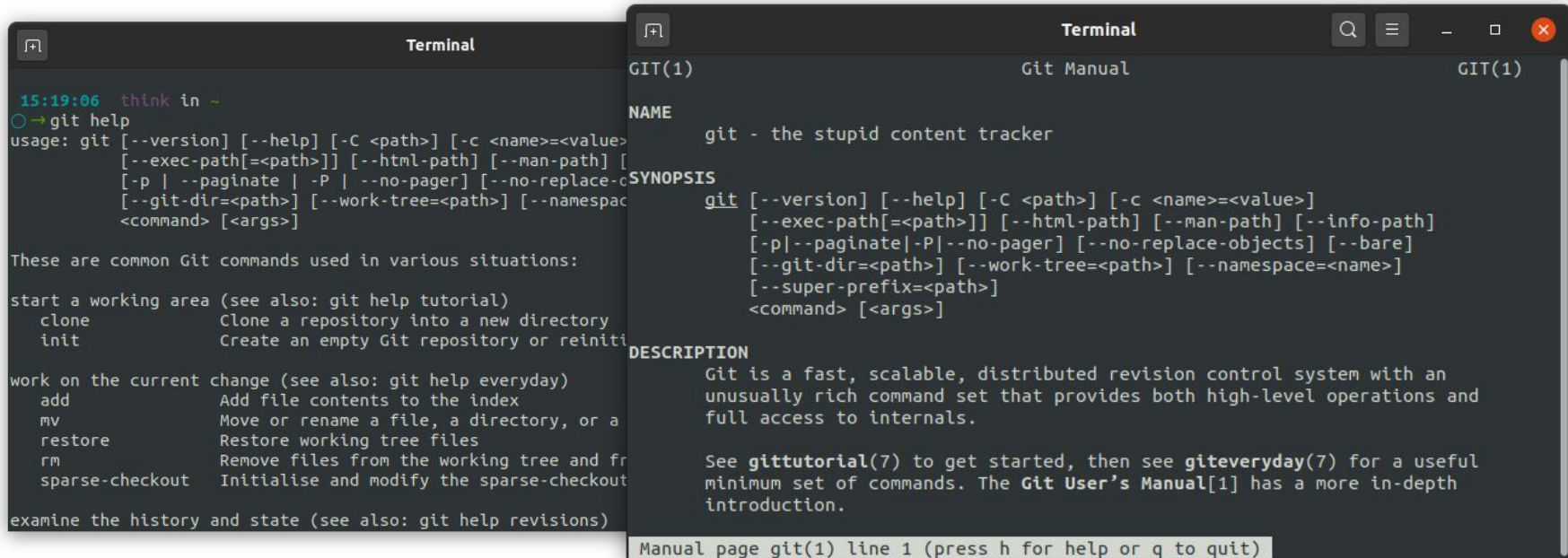
start a working area (see also: git help tutorial)
  clone                Clone a repository into a new directory
  init                 Create an empty Git repository or reinitialise an existing one

work on the current change (see also: git help everyday)
  add                  Add file contents to the index
  mv                   Move or rename a file, a directory, or a symlink
  restore              Restore working tree files
  rm                   Remove files from the working tree and from the index
  sparse-checkout      Initialise and modify the sparse-checkout

examine the history and state (see also: git help revisions)
```

GIT HELP

It's always good to know we can ask for help, `git help` is there for us
And so is `man git`



```
15:19:06 think in ~
➜ git help
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
      [--exec-path[=<path>]] [--html-path] [--man-path] [
      [-p | --paginate | -P | --no-pager] [--no-replace-objects]
      [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
      <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
  clone                Clone a repository into a new directory
  init                Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
  add                 Add file contents to the index
  mv                  Move or rename a file, a directory, or a symlink
  restore             Restore working tree files
  rm                  Remove files from the working tree and the index
  sparse-checkout      Initialise and modify the sparse-checkout

examine the history and state (see also: git help revisions)
```

```
GIT(1)                                Git Manual                                GIT(1)

NAME
  git - the stupid content tracker

SYNOPSIS
  git [--version] [--help] [-C <path>] [-c <name>=<value>]
    [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
    [-p|--paginate|-P|--no-pager] [--no-replace-objects] [--bare]
    [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
    [--super-prefix=<path>]
    <command> [<args>]

DESCRIPTION
  Git is a fast, scalable, distributed revision control system with an
  unusually rich command set that provides both high-level operations and
  full access to internals.

  See gittutorial\(7\) to get started, then see giteveryday\(7\) for a useful
  minimum set of commands. The Git User's Manual\[1\] has a more in-depth
  introduction.

Manual page git(1) line 1 (press h for help or q to quit)
```

GIT INIT

This is what we start with.

With `git init` command, we'll create `demo_repo`

```
00:22:13 think in ~/github
○→git init demo_repo
Initialised empty Git repository in /home/kzkedzierska/github/demo_repo/.git/

00:42:59 think in ~/github
○→ll demo_repo/
total 12K
drwxrwxr-x  3 kzkedzierska kzkedzierska 4.0K Nov 10 00:42 ./
drwxrwxr-x 14 kzkedzierska kzkedzierska 4.0K Nov 10 00:42 ../
drwxrwxr-x  7 kzkedzierska kzkedzierska 4.0K Nov 10 00:42 .git/
```

.GIT/

With this we also created this folder



```
00:43:09 think in ~/github
○→ll demo_repo/.git/
total 40K
drwxrwxr-x 7 kzkezdierska kzkezdierska 4.0K Nov 10 00:42 ./
drwxrwxr-x 3 kzkezdierska kzkezdierska 4.0K Nov 10 00:42 ../
drwxrwxr-x 2 kzkezdierska kzkezdierska 4.0K Nov 10 00:42 branches/
-rw-rw-r-- 1 kzkezdierska kzkezdierska 92 Nov 10 00:42 config
-rw-rw-r-- 1 kzkezdierska kzkezdierska 73 Nov 10 00:42 description
-rw-rw-r-- 1 kzkezdierska kzkezdierska 23 Nov 10 00:42 HEAD
drwxrwxr-x 2 kzkezdierska kzkezdierska 4.0K Nov 10 00:42 hooks/
drwxrwxr-x 2 kzkezdierska kzkezdierska 4.0K Nov 10 00:42 info/
drwxrwxr-x 4 kzkezdierska kzkezdierska 4.0K Nov 10 00:42 objects/
drwxrwxr-x 4 kzkezdierska kzkezdierska 4.0K Nov 10 00:42 refs/
```

CREATING CONTENT

```
02:15:19 think in ~/github/demo_repo
± |master ✓| →ll
total 12K
drwxrwxr-x  3 kzkedzierska kzkedzierska 4.0K Nov 10 02:14 ./
drwxrwxr-x 14 kzkedzierska kzkedzierska 4.0K Nov 10 01:29 ../
drwxrwxr-x  7 kzkedzierska kzkedzierska 4.0K Nov 10 02:15 .git/

02:15:36 think in ~/github/demo_repo
± |master ✓| →touch README

02:15:41 think in ~/github/demo_repo
± |master ? :1 ✗| →ll
total 12K
drwxrwxr-x  3 kzkedzierska kzkedzierska 4.0K Nov 10 02:15 ./
drwxrwxr-x 14 kzkedzierska kzkedzierska 4.0K Nov 10 01:29 ../
drwxrwxr-x  7 kzkedzierska kzkedzierska 4.0K Nov 10 02:15 .git/
-rw-rw-r--  1 kzkedzierska kzkedzierska   0 Nov 10 02:15 README
```

GIT STATUS

With `git status` command we can monitor our repository

```
02:16:53 think in ~/github/demo_repo
± |master ? :1 X| → git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    README

nothing added to commit but untracked files present (use "git add" to track)
```

GIT ADD

`git add` allows us to add

```
02:17:31 think in ~/github/demo_repo
± |master ?:1 X| →git add README

02:17:34 think in ~/github/demo_repo
± |master S:1 X| →git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   README
```


GIT ADD

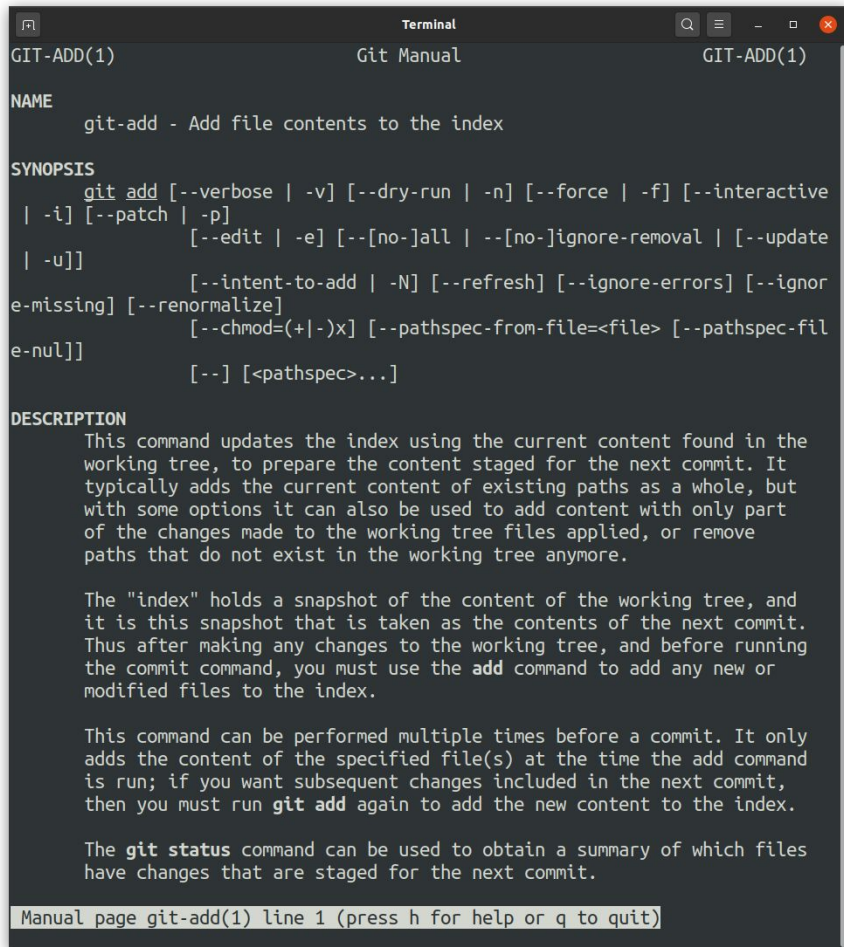
To check out all options see
`man git add`

`git add` supports basic regex
syntax, meaning:

`git add README.md` will add
README.md file to the list of tracked files

`git add *.txt` will add all txt files

`git add .` will add all non hidden files
in the current working directory



```
Terminal
GIT-ADD(1)                               Git Manual                               GIT-ADD(1)

NAME
    git-add - Add file contents to the index

SYNOPSIS
    git add [--verbose | -v] [--dry-run | -n] [--force | -f] [--interactive
    | -i] [--patch | -p]
        [--edit | -e] [--[no-]all | --[no-]ignore-removal | [--update
    | -u]]
        [--intent-to-add | -N] [--refresh] [--ignore-errors] [--ignor
    e-missing] [--renormalize]
        [--chmod=(+|-)x] [--pathspec-from-file=<file>] [--pathspec-fil
    e-nul]]
        [--] [<pathspec>...]

DESCRIPTION
    This command updates the index using the current content found in the
    working tree, to prepare the content staged for the next commit. It
    typically adds the current content of existing paths as a whole, but
    with some options it can also be used to add content with only part
    of the changes made to the working tree files applied, or remove
    paths that do not exist in the working tree anymore.

    The "index" holds a snapshot of the content of the working tree, and
    it is this snapshot that is taken as the contents of the next commit.
    Thus after making any changes to the working tree, and before running
    the commit command, you must use the add command to add any new or
    modified files to the index.

    This command can be performed multiple times before a commit. It only
    adds the content of the specified file(s) at the time the add command
    is run; if you want subsequent changes included in the next commit,
    then you must run git add again to add the new content to the index.

    The git status command can be used to obtain a summary of which files
    have changes that are staged for the next commit.

Manual page git-add(1) line 1 (press h for help or q to quit)
```

GIT COMMIT

`git commit` saves changes to a particular commit

`git commit -m "Commit message here"` avoids vim/vi editing

```
02:17:38 think in ~/github/demo_repo
± |master S:1 X| →git commit -m "initial commit"
[master (root-commit) 402b5a6] initial commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 README

02:22:33 think in ~/github/demo_repo
± |master ✓| →git status
On branch master
nothing to commit, working tree clean
```

GIT COMMIT MESSAGE

Make sure message is human readable!



	COMMENT	DATE
○	CREATED MAIN LOOP & TIMING CONTROL	14 HOURS AGO
○	ENABLED CONFIG FILE PARSING	9 HOURS AGO
○	MISC BUGFIXES	5 HOURS AGO
○	CODE ADDITIONS/EDITS	4 HOURS AGO
○	MORE CODE	4 HOURS AGO
○	HERE HAVE CODE	4 HOURS AGO
○	AAAAAAA	3 HOURS AGO
○	ADKFJSLKDFJSDKLFT	3 HOURS AGO
○	MY HANDS ARE TYPING WORDS	2 HOURS AGO
○	HAAAAAAAANDS	2 HOURS AGO

AS A PROJECT DRAGS ON, MY GIT COMMIT
MESSAGES GET LESS AND LESS INFORMATIVE.

Source: [xkcd - Git Commit](#)

GIT COMMIT

`git commit -am "Commit message here"` automatically commits all changes to already **tracked files**

To **add new files and commit** the changes run

`git add . && git commit -m "Commit message here"`

For more look up `man git commit` and linked cheat sheets.

GIT COMMIT --AMEND

`git commit --amend` let's you fix your last commit's message

`git add && git commit --amend` let's you change little things still under last commit

GIT REMOVE

`git rm path_to_file` allows us to remove a file that we're already tracking

`git rm --cached path_to_file` this is helpful when we forget to use `git rm` and instead just remove a file with `rm`

GIT REMOVE

```
01:54:01 think in ~/github/demo_repo
± |master ✓| →git rm this_other_file.txt
rm 'this_other_file.txt'

01:54:33 think in ~/github/demo_repo
± |master S:1 ✗| →git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        deleted:    this_other_file.txt

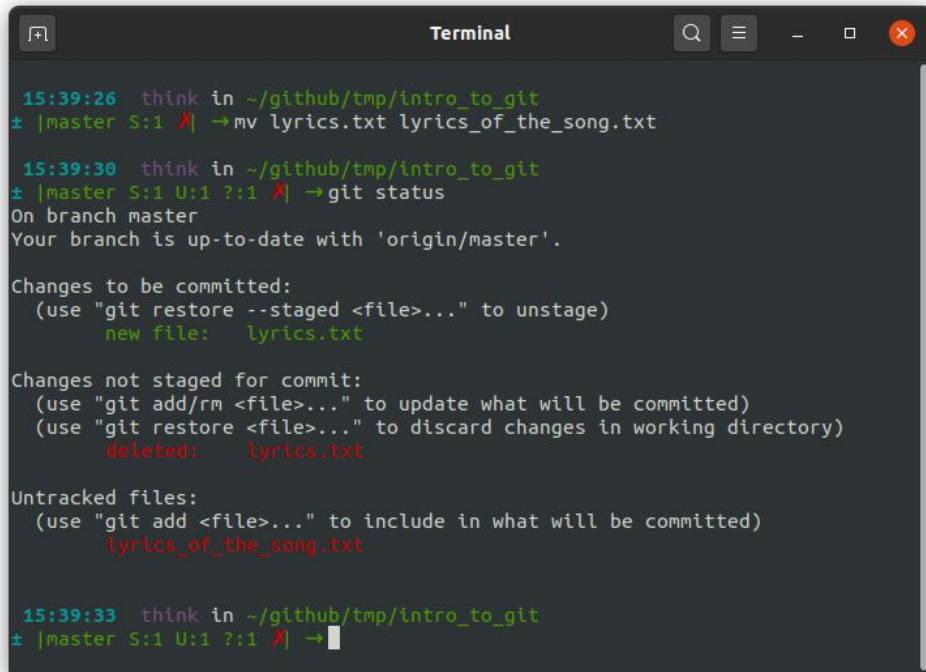
01:54:47 think in ~/github/demo_repo
± |master S:1 ✗| →git add . && git commit -m "removed some files"
[master 5e6e695] removed some files
1 file changed, 0 insertions(+), 0 deletions(-)
delete mode 100644 this_other_file.txt
```

GIT MOVE

```
git mv old_name new_name
```

change name of the file or move them to a different directory

If we use `mv` instead of `git mv` we will run into problems, as git won't know that that is the same file instead of 2.

A terminal window titled "Terminal" with a search icon, menu icon, and window control buttons. It shows a series of commands and their outputs. The first command is `git mv lyrics.txt lyrics_of_the_song.txt`, which is successful. The second command is `git status`, which shows that `lyrics.txt` is staged for deletion and `lyrics_of_the_song.txt` is staged for addition. The terminal output is as follows:

```
15:39:26 think in ~/github/tmp/intro_to_git
± |master S:1 X| → mv lyrics.txt lyrics_of_the_song.txt

15:39:30 think in ~/github/tmp/intro_to_git
± |master S:1 U:1 ?:1 X| → git status
On branch master
Your branch is up-to-date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   lyrics.txt

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    deleted:    lyrics.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    lyrics_of_the_song.txt

15:39:33 think in ~/github/tmp/intro_to_git
± |master S:1 U:1 ?:1 X| →
```


.GITIGNORE

This is a file in which we can store a list of files we don't want to be tracked.

Why would we do it? Exclude the data files (GitHub limits: 100 MB), private config files etc.

.GITIGNORE

```
./  
../  
empty_dir/  
.git/  
.gitignore  
nope1.txt  
nope2.txt  
not_this_file.txt  
README  
this_file.txt  
this_other_file.txt
```

```
16:42:50 think in ~/github/demo_repo  
± |master ?:3 X| → head .gitignore  
not_this_file.txt  
nope*
```

```
16:42:54 think in ~/github/demo_repo  
± |master ?:3 X| → git add .
```

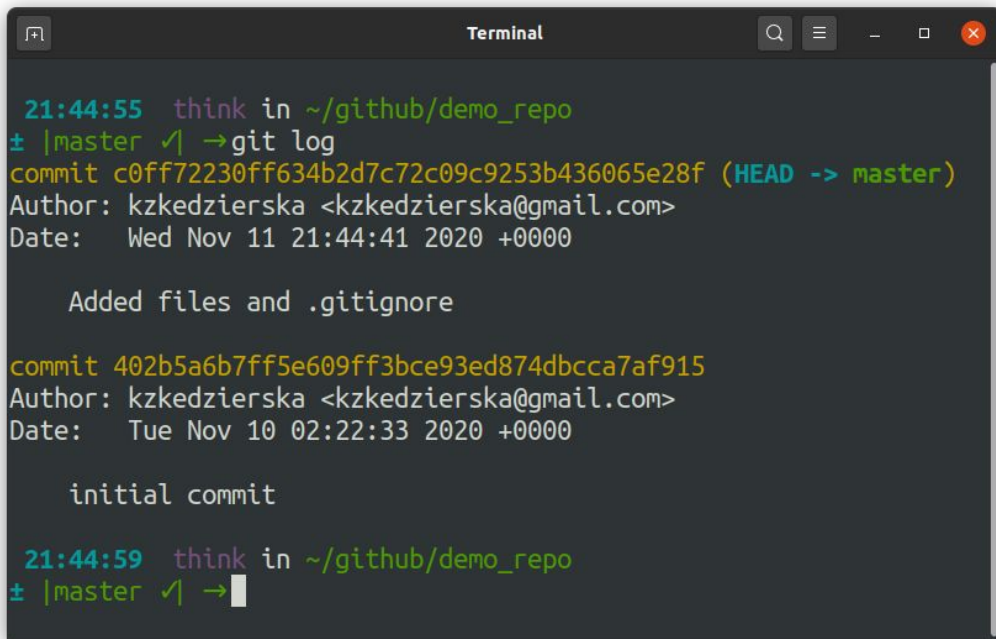
```
16:43:00 think in ~/github/demo_repo  
± |master S:3 X| → git status  
On branch master  
Changes to be committed:  
  (use "git restore --staged <file>..." to unstage)  
    new file:   .gitignore  
    new file:   this_file.txt  
    new file:   this_other_file.txt
```

GIT LOG

`git log` shows the
history of commits

`git log --oneline`

`git shortlog`

A terminal window titled "Terminal" with a dark background. It shows the output of the 'git log' command. The first commit is 'commit c0ff72230ff634b2d7c72c09c9253b436065e28f (HEAD -> master)' by 'kzkedzierska' on 'Wed Nov 11 21:44:41 2020'. The commit message is 'Added files and .gitignore'. The second commit is 'commit 402b5a6b7ff5e609ff3bce93ed874dbcca7af915' by 'kzkedzierska' on 'Tue Nov 10 02:22:33 2020'. The commit message is 'initial commit'. The terminal shows the user 'think' in the directory '~/github/demo_repo' at the 'master' branch.

```
21:44:55 think in ~/github/demo_repo
± |master ✓| →git log
commit c0ff72230ff634b2d7c72c09c9253b436065e28f (HEAD -> master)
Author: kzkedzierska <kzkedzierska@gmail.com>
Date:   Wed Nov 11 21:44:41 2020 +0000

    Added files and .gitignore

commit 402b5a6b7ff5e609ff3bce93ed874dbcca7af915
Author: kzkedzierska <kzkedzierska@gmail.com>
Date:   Tue Nov 10 02:22:33 2020 +0000

    initial commit

21:44:59 think in ~/github/demo_repo
± |master ✓| →
```

REMOTES, I.E. THE PLACE
WERE CODE LIVES

GIT CLIENTS - (IMO) TWO MAIN PLAYERS

Git can be run entirely locally, but to share the code, collaborate you need a server (remote).

GitHub: <https://github.com/>

Bitbucket: <https://bitbucket.org/product>

Biggest differences:

- no limit on file size in Bitbucket (?)
- unlimited private repos on GitHub

GIT CLONE

`git clone` allows to copy existing repository from external source

bit.ly/Git_repo

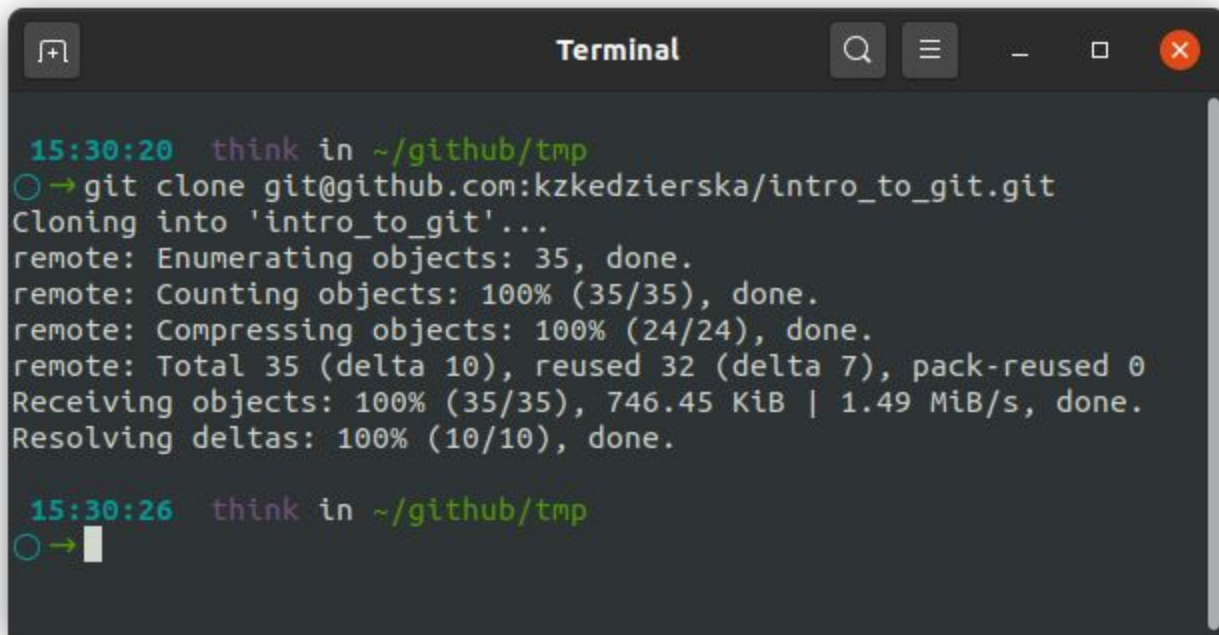
```
git clone git@github.com:kzkedzierska/intro\_to\_git.git
```

or

```
git clone https://github.com/kzkedzierska/intro\_to\_git.git
```

GIT CLONE

git clone allows to copy existing repository from external source

A terminal window titled "Terminal" with standard macOS window controls (search, menu, zoom, close). The terminal shows a user named "think" in the directory "~/github/tmp". They execute the command "git clone git@github.com:kzkdzierska/intro_to_git.git". The output shows the cloning process: enumerating 35 objects, counting 100%, compressing 100%, and receiving 746.45 KiB at 1.49 MiB/s. The command completes successfully, and the prompt returns to the user.

```
15:30:20 think in ~/github/tmp
○→git clone git@github.com:kzkdzierska/intro_to_git.git
Cloning into 'intro_to_git'...
remote: Enumerating objects: 35, done.
remote: Counting objects: 100% (35/35), done.
remote: Compressing objects: 100% (24/24), done.
remote: Total 35 (delta 10), reused 32 (delta 7), pack-reused 0
Receiving objects: 100% (35/35), 746.45 KiB | 1.49 MiB/s, done.
Resolving deltas: 100% (10/10), done.

15:30:26 think in ~/github/tmp
○→
```

BRANCH EARLY, BRANCH
OFTEN

GIT BRANCH & GIT CHECKOUT

`git branch <branch_name>` create a branch named *branch_name*

`git checkout <branch_name>` switch to a branch named *branch_name*

`git checkout -b <branch_name>` switch to a NEW branch *branch_name*

`git checkout <other_branch> && git branch -d <branch_name>`
delete a *branch_name*

GIT BRANCH & GIT CHECKOUT

```
Terminal

22:33:30 think in ~/github/intro_to_git
* |master ✓| → git checkout -b exercises
Switched to a new branch 'exercises'

22:33:33 think in ~/github/intro_to_git
* |exercises ✓| → touch only_on_exercises_branch.txt

22:33:36 think in ~/github/intro_to_git
* |exercises ? :1 A| → ll
total 868K
drwxrwxr-x 7 kzkedzierska kzkedzierska 4.0K Nov 11 22:33 ./
drwxrwxr-x 14 kzkedzierska kzkedzierska 4.0K Nov 11 22:13 ../
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 14:50 01_simple-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:36 03_branch-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:37 03_easy-merge-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:41 04_harder-merge-exercise/
-rw-rw-r-- 1 kzkedzierska kzkedzierska 1.1K Nov 10 03:00 code_snippet.sh
drwxrwxr-x 8 kzkedzierska kzkedzierska 4.0K Nov 11 22:33 .git/
-rw-rw-r-- 1 kzkedzierska kzkedzierska 16 Nov 11 16:39 .gitignore
-rw-rw-r-- 1 kzkedzierska kzkedzierska 0 Nov 11 16:38 hidden_file.txt
-rw-rw-r-- 1 kzkedzierska kzkedzierska 5.5K Nov 10 18:11 How_to_prepare.md
-rw-rw-r-- 1 kzkedzierska kzkedzierska 0 Nov 11 22:33 only_on_exercises_branch.tx
t
-rw-rw-r-- 1 kzkedzierska kzkedzierska 997 Nov 10 16:51 README.md
-rw-rw-r-- 1 kzkedzierska kzkedzierska 817K Nov 9 22:09 slides.pdf

22:33:37 think in ~/github/intro_to_git
* |exercises ? :1 A| → git add . && git commit -m "added secret file"
[exercises cf0a589] added secret file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 only_on_exercises_branch.txt
```

GIT BRANCH & GIT CHECKOUT

```
Terminal

22:33:30 think in ~/github/intro_to_git
# |master ✓| → git checkout -b exercises
Switched to a new branch 'exercises'

22:33:33 think in ~/github/intro_to_git
# |exercises ✓| → touch only_on_exercises_branch.txt

22:33:36 think in ~/github/intro_to_git
# |exercises ? :1 A| → ll
total 868K
drwxrwxr-x 7 kzkedzierska kzkedzierska 4.0K Nov 11 22:33 ./
drwxrwxr-x 14 kzkedzierska kzkedzierska 4.0K Nov 11 22:13 ../
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 14:50 01_simple-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:36 03_branch-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:37 03_easy-merge-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:41 04_harder-merge-exercise/
-rw-rw-r-- 1 kzkedzierska kzkedzierska 1.1K Nov 10 03:00 code_snippet.sh
drwxrwxr-x 8 kzkedzierska kzkedzierska 4.0K Nov 11 22:33 .git/
-rw-rw-r-- 1 kzkedzierska kzkedzierska 16 Nov 11 16:39 .gitignore
-rw-rw-r-- 1 kzkedzierska kzkedzierska 0 Nov 11 16:38 hidden_file.txt
-rw-rw-r-- 1 kzkedzierska kzkedzierska 5.5K Nov 10 18:11 How_to_prepare.md
-rw-rw-r-- 1 kzkedzierska kzkedzierska 0 Nov 11 22:33 only_on_exercises_branch.txt
-rw-rw-r-- 1 kzkedzierska kzkedzierska 997 Nov 10 16:51 README.md
-rw-rw-r-- 1 kzkedzierska kzkedzierska 817K Nov 9 22:09 slides.pdf

22:33:37 think in ~/github/intro_to_git
# |exercises ? :1 A| → git add . && git commit -m "added secret file"
[exercises cf0a589] added secret file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 only_on_exercises_branch.txt
```

```
Terminal

22:33:43 think in ~/github/intro_to_git
# |exercises ✓| → git checkout master
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.

22:33:55 think in ~/github/intro_to_git
# |master ✓| → ll
total 868K
drwxrwxr-x 7 kzkedzierska kzkedzierska 4.0K Nov 11 22:33 ./
drwxrwxr-x 14 kzkedzierska kzkedzierska 4.0K Nov 11 22:13 ../
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 14:50 01_simple-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:36 03_branch-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:37 03_easy-merge-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:41 04_harder-merge-exercise/
-rw-rw-r-- 1 kzkedzierska kzkedzierska 1.1K Nov 10 03:00 code_snippet.sh
drwxrwxr-x 8 kzkedzierska kzkedzierska 4.0K Nov 11 22:33 .git/
-rw-rw-r-- 1 kzkedzierska kzkedzierska 16 Nov 11 16:39 .gitignore
-rw-rw-r-- 1 kzkedzierska kzkedzierska 0 Nov 11 16:38 hidden_file.txt
-rw-rw-r-- 1 kzkedzierska kzkedzierska 5.5K Nov 10 18:11 How_to_prepare.md
-rw-rw-r-- 1 kzkedzierska kzkedzierska 997 Nov 10 16:51 README.md
-rw-rw-r-- 1 kzkedzierska kzkedzierska 817K Nov 9 22:09 slides.pdf

22:33:56 think in ~/github/intro_to_git
# |master ✓| →
```

GIT MERGE

```
git merge <branch_name>
```

merges the `branch_name`
into current branch

We experimented, tested
new code and are ready
to merge it into another
branch, master for
example

```
Terminal

22:33:56 think in ~/github/intro_to_git
± |master ✓| → git merge exercises
Updating fee7142..cf0a589
Fast-forward
 only_on_exercises_branch.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 only_on_exercises_branch.txt

22:39:53 think in ~/github/intro_to_git
± |master ↑1 ✓| → ll
total 868K
drwxrwxr-x 7 kzkedzierska kzkedzierska 4.0K Nov 11 22:39 ./
drwxrwxr-x 14 kzkedzierska kzkedzierska 4.0K Nov 11 22:13 ../
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 14:50 01_simple-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:36 03_branch-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:37 03_easy-merge-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:41 04_harder-merge-exercise/
-rw-rw-r-- 1 kzkedzierska kzkedzierska 1.1K Nov 10 03:00 code_snippet.sh
drwxrwxr-x 8 kzkedzierska kzkedzierska 4.0K Nov 11 22:39 .git/
-rw-rw-r-- 1 kzkedzierska kzkedzierska 16 Nov 11 16:39 .gitignore
-rw-rw-r-- 1 kzkedzierska kzkedzierska 0 Nov 11 16:38 hidden_file.txt
-rw-rw-r-- 1 kzkedzierska kzkedzierska 5.5K Nov 10 18:11 How_to_prepare.md
-rw-rw-r-- 1 kzkedzierska kzkedzierska 0 Nov 11 22:39 only_on_exercises_branch.txt
t
-rw-rw-r-- 1 kzkedzierska kzkedzierska 997 Nov 10 16:51 README.md
-rw-rw-r-- 1 kzkedzierska kzkedzierska 817K Nov 9 22:09 slides.pdf

22:39:58 think in ~/github/intro_to_git
± |master ↑1 ✓| → git branch -d exercises
Deleted branch exercises (was cf0a589).

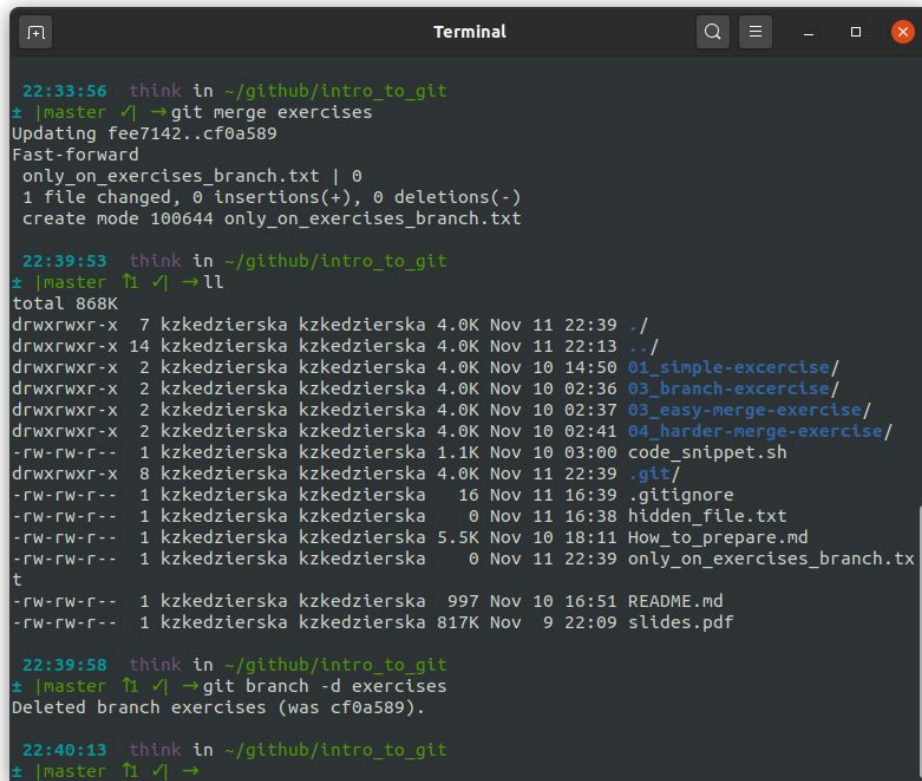
22:40:13 think in ~/github/intro_to_git
± |master ↑1 ✓| →
```

GIT MERGE

```
git merge <branch_name>
```

merges the `branch_name`
into current branch

We experimented, tested
new code and are ready
to merge it into another
branch, master for
example

A terminal window titled "Terminal" with a search icon, menu icon, and window control buttons. It shows a series of commands and their outputs. The first command is `git merge exercises`, which updates the master branch with the changes from the 'exercises' branch. The output shows that one file, 'only_on_exercises_branch.txt', was created. The second command is `git branch -d exercises`, which deletes the 'exercises' branch. The output shows that the branch was successfully deleted.

```
22:33:56 think in ~/github/intro_to_git
± |master ✓| → git merge exercises
Updating fee7142..cf0a589
Fast-forward
  only_on_exercises_branch.txt | 0
  1 file changed, 0 insertions(+), 0 deletions(-)
  create mode 100644 only_on_exercises_branch.txt

22:39:53 think in ~/github/intro_to_git
± |master ↑1 ✓| → ll
total 868K
drwxrwxr-x 7 kzkedzierska kzkedzierska 4.0K Nov 11 22:39 ./
drwxrwxr-x 14 kzkedzierska kzkedzierska 4.0K Nov 11 22:13 ../
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 14:50 01_simple-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:36 03_branch-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:37 03_easy-merge-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:41 04_harder-merge-exercise/
-rw-rw-r-- 1 kzkedzierska kzkedzierska 1.1K Nov 10 03:00 code_snippet.sh
drwxrwxr-x 8 kzkedzierska kzkedzierska 4.0K Nov 11 22:39 .git/
-rw-rw-r-- 1 kzkedzierska kzkedzierska 16 Nov 11 16:39 .gitignore
-rw-rw-r-- 1 kzkedzierska kzkedzierska 0 Nov 11 16:38 hidden_file.txt
-rw-rw-r-- 1 kzkedzierska kzkedzierska 5.5K Nov 10 18:11 How_to_prepare.md
-rw-rw-r-- 1 kzkedzierska kzkedzierska 0 Nov 11 22:39 only_on_exercises_branch.txt
t
-rw-rw-r-- 1 kzkedzierska kzkedzierska 997 Nov 10 16:51 README.md
-rw-rw-r-- 1 kzkedzierska kzkedzierska 817K Nov 9 22:09 slides.pdf

22:39:58 think in ~/github/intro_to_git
± |master ↑1 ✓| → git branch -d exercises
Deleted branch exercises (was cf0a589).

22:40:13 think in ~/github/intro_to_git
± |master ↑1 ✓| →
```


GIT PULL & GIT PUSH



EXERCISES! :)

LOOK FOR EXERCISES IN THE REPOSITORY

bit.ly/Git_repo

	kzkedzierska change directory names for consistency	170c8f0 13 seconds ago	🕒 11 commits
📁	01_simple_excercise	change directory names for consistency	13 seconds ago
📁	02_ammend_commit	Added the exercises	8 minutes ago
📁	03_branch_excercise	change directory names for consistency	13 seconds ago
📄	.gitignore	updated instructions and added gitignore	9 hours ago
📄	How_to_prepare.md	updated instructions and added gitignore	9 hours ago
📄	README.md	change directory names for consistency	13 seconds ago
📄	code_snippet.sh	Created the backbone of the exercises	2 days ago
📄	slides.pdf	Added slides	11 months ago

SUMMARY

REMEMBER THE BASICS, GOOGLE FOR HELP WHEN NEEDED

init, clone

status

add, rm, mv

log

commit

push

branch, checkout

merge

push, pull

REMEMBER THE BASICS, GOOGLE FOR HELP WHEN NEEDED

init, **clone**

add, **rm**, **mv**

commit

push

branch, **checkout**

merge

push, **pull**

status

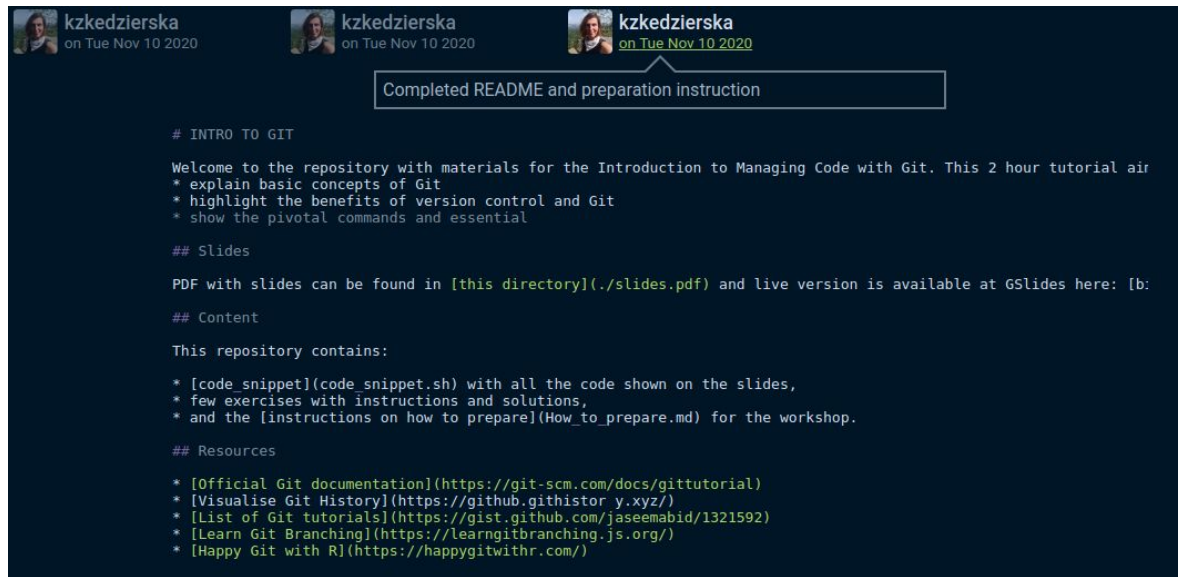
log

Other interesting ones

show, **diff**

reset, **revert**

LET'S SEE COMMIT HISTORY IN ACTION



```
# INTRO TO GIT

Welcome to the repository with materials for the Introduction to Managing Code with Git. This 2 hour tutorial air
* explain basic concepts of Git
* highlight the benefits of version control and Git
* show the pivotal commands and essential

## Slides

PDF with slides can be found in [this directory](./slides.pdf) and live version is available at GSlides here: [b:

## Content

This repository contains:

* [code_snippet](code_snippet.sh) with all the code shown on the slides,
* few exercises with instructions and solutions,
* and the [instructions on how to prepare](How_to_prepare.md) for the workshop.

## Resources

* [Official Git documentation](https://git-scm.com/docs/gittutorial)
* [Visualise Git History](https://github.githubior y.xyz/)
* [List of Git tutorials](https://gist.github.com/jaseemabid/1321592)
* [Learn Git Branching](https://learngitbranching.js.org/)
* [Happy Git with R](https://happygitwithr.com/)
```

github.githubior y.xyz/kzkedzierska/intro_to_git/blob/master/README.md

THANKS!

Big shout out to **Duncan Parkes**!

Some resources:

- [Official Git tutorial](#)
- [Visualise Git History](#)
- [List of Git tutorials](#)
- [Learn Git Branching](#)
- [Happy Git with R](#)

Get in touch!

kasia@well.ox.ac.uk

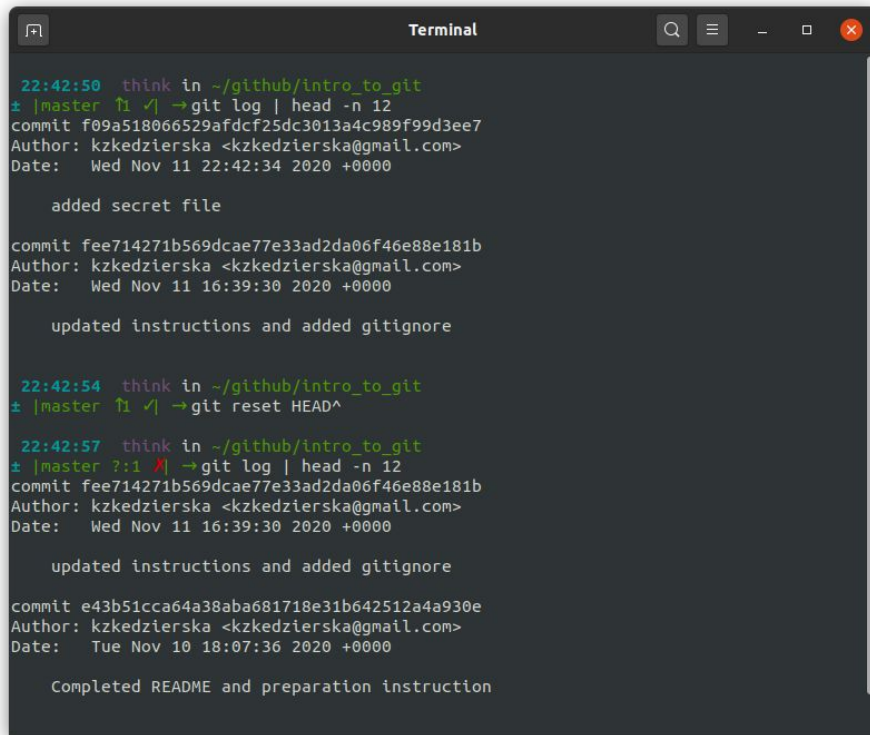
duncan@well.ox.ac.uk



Read about ducks in programming [here](#). :)

IF WE HAVE SOME TIME...

GIT RESET - WHEN WE WANT TO GO BACK



```
22:42:50 think in ~/github/intro_to_git
± |master ↑1 ✓| →git log | head -n 12
commit f09a518066529afdcf25dc3013a4c989f99d3ee7
Author: kzkedzierska <kzkedzierska@gmail.com>
Date: Wed Nov 11 22:42:34 2020 +0000

    added secret file

commit fee714271b569dcae77e33ad2da06f46e88e181b
Author: kzkedzierska <kzkedzierska@gmail.com>
Date: Wed Nov 11 16:39:30 2020 +0000

    updated instructions and added gitignore

22:42:54 think in ~/github/intro_to_git
± |master ↑1 ✓| →git reset HEAD^

22:42:57 think in ~/github/intro_to_git
± |master ? :1 X| →git log | head -n 12
commit fee714271b569dcae77e33ad2da06f46e88e181b
Author: kzkedzierska <kzkedzierska@gmail.com>
Date: Wed Nov 11 16:39:30 2020 +0000

    updated instructions and added gitignore

commit e43b51cca64a38aba681718e31b642512a4a930e
Author: kzkedzierska <kzkedzierska@gmail.com>
Date: Tue Nov 10 18:07:36 2020 +0000

    Completed README and preparation instruction
```


GIT RESET - WHEN WE WANT TO GO BACK

```
22:42:50 think in ~/github/intro_to_git
± |master ↗ ↵| →git log | head -n 12
commit f09a518066529afdcf25dc3013a4c989f99d3ee7
Author: kzkedzierska <kzkedzierska@gmail.com>
Date: Wed Nov 11 22:42:34 2020 +0000

    added secret file

commit fee714271b569dcae77e33ad2da06f46e88e181b
Author: kzkedzierska <kzkedzierska@gmail.com>
Date: Wed Nov 11 16:39:30 2020 +0000

    updated instructions and added gitignore

22:42:54 think in ~/github/intro_to_git
± |master ↗ ↵| →git reset HEAD^

22:42:57 think in ~/github/intro_to_git
± |master ? :1 ↗ ↵| →git log | head -n 12
commit fee714271b569dcae77e33ad2da06f46e88e181b
Author: kzkedzierska <kzkedzierska@gmail.com>
Date: Wed Nov 11 16:39:30 2020 +0000

    updated instructions and added gitignore

commit e43b51cca64a38aba681718e31b642512a4a930e
Author: kzkedzierska <kzkedzierska@gmail.com>
Date: Tue Nov 10 18:07:36 2020 +0000

    Completed README and preparation instruction
```

```
22:43:02 think in ~/github/intro_to_git
± |master ? :1 ↗ ↵| →ll
total 868K
drwxrwxr-x 7 kzkedzierska kzkedzierska 4.0K Nov 11 22:42 ./
drwxrwxr-x 14 kzkedzierska kzkedzierska 4.0K Nov 11 22:13 ../
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 14:50 01_simple-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:36 03_branch-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:37 03_easy-merge-exercise/
drwxrwxr-x 2 kzkedzierska kzkedzierska 4.0K Nov 10 02:41 04_harder-merge-exercise/
-rw-rw-r-- 1 kzkedzierska kzkedzierska 1.1K Nov 10 03:00 code_snippet.sh
drwxrwxr-x 8 kzkedzierska kzkedzierska 4.0K Nov 11 22:43 .git/
-rw-rw-r-- 1 kzkedzierska kzkedzierska 16 Nov 11 16:39 .gitignore
-rw-rw-r-- 1 kzkedzierska kzkedzierska 0 Nov 11 16:38 hidden_file.txt
-rw-rw-r-- 1 kzkedzierska kzkedzierska 5.5K Nov 10 18:11 How_to_prepare.md
-rw-rw-r-- 1 kzkedzierska kzkedzierska 0 Nov 11 22:42 only_on_exercises_branch.tx
t
-rw-rw-r-- 1 kzkedzierska kzkedzierska 997 Nov 10 16:51 README.md
-rw-rw-r-- 1 kzkedzierska kzkedzierska 817K Nov 9 22:09 slides.pdf

22:43:55 think in ~/github/intro_to_git
± |master ? :1 ↗ ↵| →rm only_on_exercises_branch.txt
rm: remove regular empty file 'only_on_exercises_branch.txt'? y
removed 'only_on_exercises_branch.txt'

22:44:10 think in ~/github/intro_to_git
```