

GIT and GitHub

Gergely Fekete

2020.02.26 - Lab Meeting

GIT

GIT is a tool to save folders , and handle versions












GitHub

GitHub is a webserver where you can save

- GIT works without GitHub
- GitHub is only one of the many git servers
- Git servers adds the ability to share your stuff

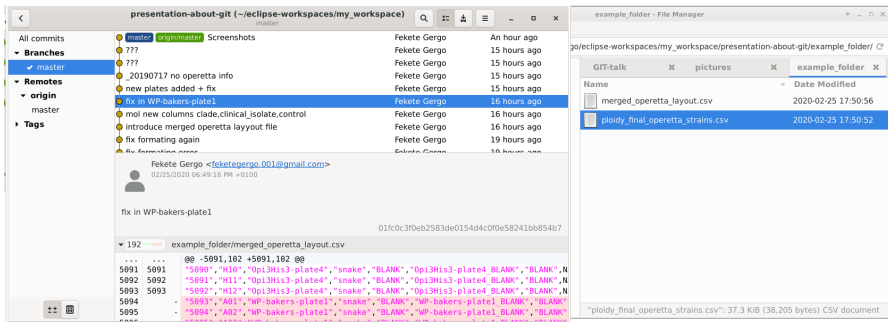
What are versions?

- Sometimes we edit a file continuously and want to keep its earlier versions

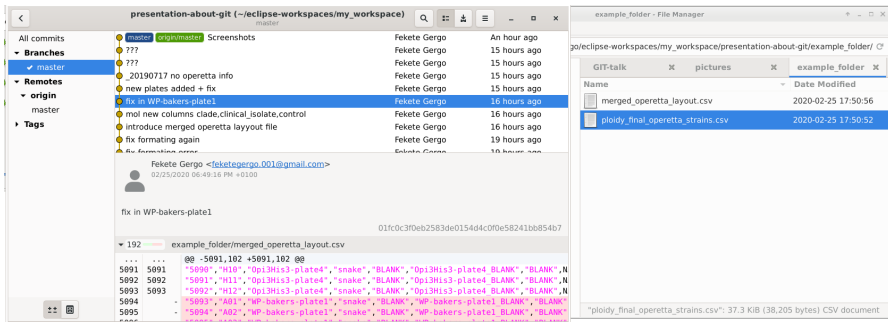
	merged_operetta_layout_20200215.csv	3.7 MiB	20
	merged_operetta_layout_20200210.csv	3.6 MiB	20
	merged_operetta_layout_20200204.csv	3.6 MiB	20
	merged_operetta_layout_20200115.csv	3.2 MiB	20
	merged_operetta_layout_20200113-tmp.xls	4.0 MiB	20
	merged_operetta_layout_20200113.csv	2.9 MiB	20
	merged_operetta_layout_20191217-tmp.xlsx	1018.0 KiB	20
	merged_operetta_layout_20191217.csv	2.4 MiB	20
	merged_operetta_layout_20191216.csv	2.4 MiB	20
	merged_operetta_layout_20191213.csv	2.4 MiB	20
	merged_operetta_layout_20191129.csv	2.4 MiB	20

- the state of the art solution
 - have one file in the working directory
 - store the old versions 'hidden' in a **repository**
- What is a **repository**?
 - a simple subfolder
 - The folder name is '.git'.
 - It is a hidden folder
 - You have to start git to see the content

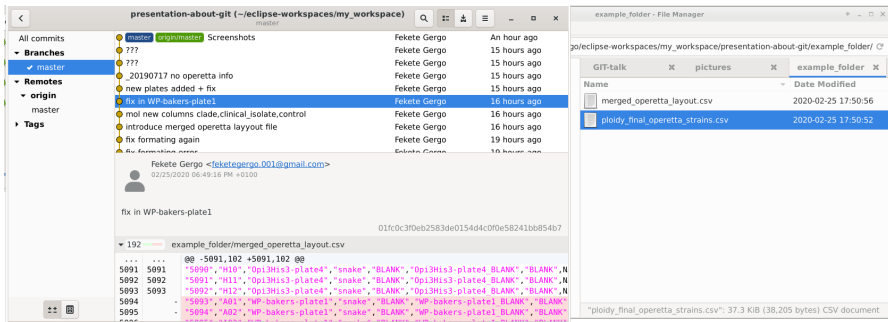
- the state of the art solution
 - have one file in the working directory
 - store the old versions 'hidden' in a **repository**
- What is a **repository**?
 - a simple subfolder
 - The folder name is `'.git'`.
 - It is a hidden folder
 - You have to start git to see the content



- normally you see only the 2 important files
- If you need the old versions you can turn on the repository browser.
- Each ball represents a previous version
- the term of the 'balls' is commit/revision/version
- You can delete files from the working directory. The repo keeps it.



- normally you see only the 2 important files
- If you need the old versions you can turn on the repository browser.
- Each ball represents a previous version
- the term of the 'balls' is commit/revision/version
- You can delete files from the working directory. The repo keeps it.



- normally you see only the 2 important files
- If you need the old versions you can turn on the repository browser.
- Each ball represents a previous version
- the term of the 'balls' is commit/revision/version
- You can delete files from the working directory. The repo keeps it.

Name	Size	Date Modified
plakdy_final_operetta_strains_190807_mod2-verGer-tmp.csv	36.8 KiB	2019-12-20 22:45:22
plakdy_final_operetta_strains_190807_mod2.csv	37.0 KiB	2019-10-23 16:35:17
plakdy_final_operetta_strains_190807_mod.csv	37.3 KiB	2019-08-07 18:40:33
merged_operetta_layout_20200224.csv	3.7 MiB	2020-02-24 16:53:11
merged_operetta_layout_20200215.csv	3.7 MiB	2020-02-15 20:21:35
merged_operetta_layout_20200210.csv	3.6 MiB	2020-02-10 13:24:00
merged_operetta_layout_20200204.csv	3.6 MiB	2020-02-04 16:04:49
merged_operetta_layout_20200115.csv	3.2 MiB	2020-01-15 15:09:45
merged_operetta_layout_20200113-tmp.xls	4.0 MiB	2020-01-14 09:14:32
merged_operetta_layout_20200113.csv	2.9 MiB	2020-01-13 18:07:00
merged_operetta_layout_20191217-tmp.xls	1018.0 KiB	2019-12-18 10:53:08
merged_operetta_layout_20191217.csv	2.4 MiB	2019-12-17 12:02:21
merged_operetta_layout_20191216.csv	2.4 MiB	2019-12-16 13:11:23
merged_operetta_layout_20191213.csv	2.4 MiB	2019-12-13 13:30:20
merged_operetta_layout_20191129.csv	2.4 MiB	2019-11-29 16:36:46
merged_operetta_layout_20190807.csv	2.4 MiB	2019-08-07 17:49:06
merged_operetta_layout_20190805.csv	2.4 MiB	2019-08-05 13:36:59
merged_operetta_layout_20190803.csv	2.4 MiB	2019-08-03 14:56:55
merged_operetta_layout_20190724.csv	2.0 MiB	2019-07-24 14:36:08
merged_operetta_layout_20190717_no_operetta_info.csv	691.8 KiB	2019-07-17 14:18:48
merged_operetta_layout_20190215.csv	1.6 MiB	2019-02-15 17:50:50
merged_operetta_layout_20181122.csv	1.6 MiB	2018-11-22 17:44:42
merged_operetta_layout_20181108.csv	1.6 MiB	2018-11-10 10:49:19
merged_operetta_layout_20180911.csv	1.3 MiB	2018-09-11 13:30:37
cellnum_perplate_genotype_170616_v5_withK015.csv	56.7 KiB	2018-11-19 10:08:57
all_strains_morphology_ploidy.csv	18.6 MiB	2019-01-16 16:33:18
all_strains_morphology2.csv	16.0 MiB	2018-10-11 14:35:53
all_strains_morphology.csv	8.3 MiB	2018-08-28 13:42:32

29 items: 94.3 MiB (98,844,291 bytes). Free space: 14.1 TiB

- working with messy folders is slower and confusing
- it causes errors
- It is waste of time and money.

Back to the top

GIT

GIT is a tool to save folders , and handle versions

- now we know what are versions
- Let's see why to save forlders instead of files

Why to save folders?

Belive me! It is a result of 35 years of evoluton and desig.

Why to save folders?

Imagine a project where are

- experimental layout file
- result files form a microscope

They belong together. It is nice to connect them.

- actually it does not save full folder. You can select some files to save together.
- The principal concept is 'comit together what belongs together'

Why to save folders?

Imagine a project where are

- experimental layout file
- result files form a microscope

They belong together. It is nice to connect them.

- actually it does not save full folder. You can select some files to save together.
- The principal concept is 'comit together what belongs together'

Why to save folders?

Imagine a project where are

- experimental layout file
- result files form a microscope

They belong together. It is nice to connect them.

- actually it does not save full folder. You can select some files to save together.
- The principal concept is 'commit together what belongs together'

Back to the top again

GIT

GIT is a tool to save folders , and handle versions

- now we know what are versions
- we understand that commit many files together is clever
- What is the GIT tool?

Back to the top again

GIT

GIT is a tool to save folders , and handle versions

- now we know what are versions
- we understand that commit many files together is clever
- What is the GIT tool?

What is the GIT tool?

- actually git is not one tool: it is a protocol/standard
- There are a lot of git programs you can install.
- Linux and Mac have preinstalled git
- Rstudio contains a git client
- every IDE contains a git client (C , JAVA, python editors ...)
- gitg (graphical UI - linux, windows, mac)
- git SMC (Windows git client)
- Git Bash (Windows git terminal)
- every IDE contains a git client (C , JAVA, python editors ...)

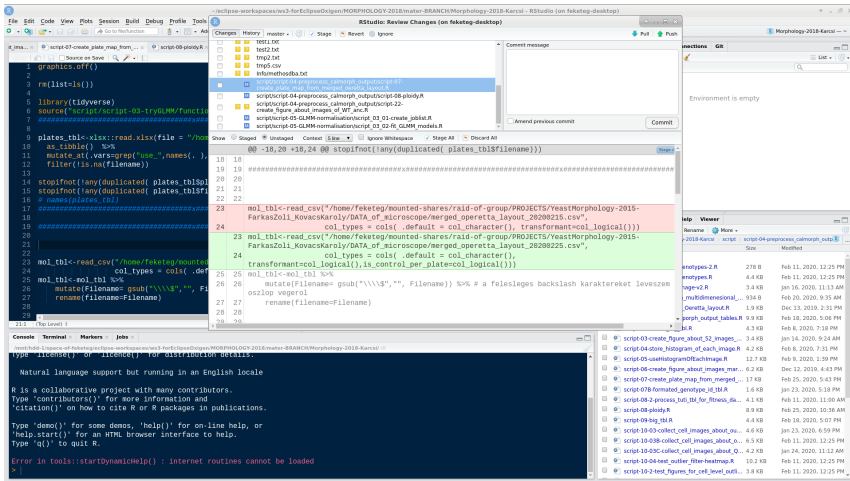
What is the GIT tool?

- actually git is not one tool: it is a protocol/standard
- There are a lot of git programs you can install.
- Linux and Mac have preinstalled git
- Rstudio contains a git client
- every IDE contains a git client (C , JAVA, python editors ...)
- gitg (graphical UI - linux, windows, mac)
- git SMC (Windows git client)
- Git Bash (Windows git terminal)
- every IDE contains a git client (C , JAVA, python editors ...)

What is the GIT tool?

- actually git is not one tool: it is a protocol/standard
- There are a lot of git programs you can install.
- Linux and Mac have preinstalled git
- Rstudio contains a git client
- every IDE contains a git client (C , JAVA, python editors ...)
- gitg (graphical UI - linux, windows, mac)
- git SMC (Windows git client)
- Git Bash (Windows git terminal)
- every IDE contains a git client (C , JAVA, python editors ...)

Let's see how to use it



Let's see how to use it

RStudio: Review Changes (on feketeg-desktop)

Changes History master Stage Revert Ignore Pull Push

test1.txt
test2.txt
tmp2.txt
tmp5.csv
Info/methods.txt
script/script-04-preprocess_calmorph_output/script-07-create_plate_map_from_merged_operetta_layout.R
script/script-04-preprocess_calmorph_output/script-08-ploidy.R
script/script-04-preprocess_calmorph_output/script-22-create_figure_about_images_of_WT_anc.R
script/script-05-GLMM-normalisation/script_03_01-create_joblist.R
script/script-05-GLMM-normalisation/script_03_02-fit_GLMM_models.R

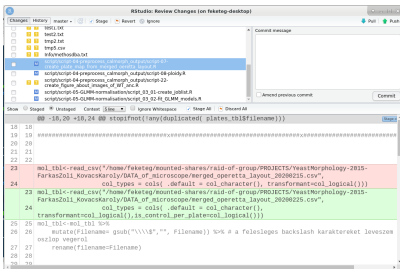
Commit message

Amend previous commit Commit

Show Staged Unstaged Context 5 line Ignore Whitespace Stage All Discard All

```
@@ -18,20 +18,24 @@ stopifnot(!any(duplicated( plates_tbl$filename)))
18 18
19 19 #####X#####X#####
20 20
21 21
22 22
23 mol_tbl<-read_csv("/home/feketeg/mounted-shares/raid-of-group/PROJECTS/YeastMorphology-2015-
24 FarkasZoli_KovacsKaroly/DATA_of_microscope/merged_operetta_layout_20200215.csv",
    col_types = cols( .default = col_character(), transformant=col_logical()))
23 mol_tbl<-read_csv("/home/feketeg/mounted-shares/raid-of-group/PROJECTS/YeastMorphology-2015-
24 FarkasZoli_KovacsKaroly/DATA_of_microscope/merged_operetta_layout_20200225.csv",
    col_types = cols( .default = col_character(),
    transformant=col_logical(),is_control_per_plate=col_logical()))
25 mol_tbl<-mol_tbl %>%
26 mutate(Filename= gsub("\\\\", "", Filename)) %>% # a felesleges backslash karaktereket leveszem
    oszlop vegerol
27 rename(filename=Filename)
28
29
```

Let's see how to use it



- select files to the stage
- unfulfilled/fulfilled files
- diff-s
- commit msg + button
- push/pull button

Terminology

- commit = save it (to the local repository)
- stage = files selected for save
- push = upload to the server
- pull = download from the server

Let's see how to use it - History

The screenshot displays the RStudio interface with the Git pane open, showing the commit history for the repository. The commit history table is as follows:

SHA	Author	Date	Subject
67002c71	feketegergo <feketegergo@desktop1.brc>	2020-02-18	A merged_oeretta_layout file datumol kovolt
c84ba60	feketegergo <feketegergo@desktop1.brc>	2020-02-11	Az xlsx mentő scriptbe betettem egy gc() hívást, mert OutOfMemory-val néha megállt.
9709daa	feketegergo <feketegergo@desktop1.brc>	2020-02-10	Bemne felejtettem egy hibát az xlsx kiíróban, azt javítottam.
aa8a8779	feketegergo <feketegergo@desktop1.brc>	2020-02-10	Nem tudom mit csináltam
c867e8d8	feketegergo <feketegergo@desktop1.brc>	2020-02-03	A script-07-create_plate_map_from_merged_oeretta_layout.R fájlaban volt egy hiba az gc() feketegergo <feketegergo@desktop1.brc>
f0d6e7a3	feketegergo <feketegergo@desktop1.brc>	2020-01-21	A mergedOerettaLayout file egy csomót változott és a kód követve a változásokat.
7017e81a	feketegergo <feketegergo@desktop1.brc>	2020-01-07	Újraírta a file-et a csomóval

The console shows the R version 3.5.2 (2018-12-13) and the RStudio version 1.2.1333 (2019-09-23). The RStudio interface also shows the Environment pane with the Global Environment and the Files pane with the file explorer.

Let's see how to use it- History

RStudio: Review Changes (on feketeg-desktop)

Changes History master • script-07-create_plate_map_from_merged_oeretta_layout.R

Subject	Author	Date	SHA
A merged_oeretta_layout file datumat koveti	feketegergo <feketegergo@desktop1.brc>	2020-02-18	67002c71
Az xlsx mentő scriptbe betettem egy gc() hívást, mert OutOfMemory-val néha megállt.	feketegergo <feketegergo@desktop1.brc>	2020-02-11	c84bade0
Benne felejtettem egy hibát az xlsx kiírásban. azt javítottam.	feketegergo <feketegergo@desktop1.brc>	2020-02-10	97709daa
nem tudom mit csináltam	feketegergo <feketegergo@desktop1.brc>	2020-02-10	aaba8779
A script-07-create_plate_map_from_merged_oeretta_layout.R fileban volt egy hiba az ger feketegergo <feketegergo@desktop1.brc>	feketegergo <feketegergo@desktop1.brc>	2020-02-03	c6e7ed8d
A mergedOerettaLayout file egy csomót változott és a kód követve a változásokat.	feketegergo <feketegergo@desktop1.brc>	2020-01-21	f0dde7a3

Commits 1 of 7

SHA: 67002c71
Author: feketegergo <feketegergo@desktop1.brc>
Date: 2020-02-18 16:08
Subject: A merged_oeretta_layout file datumat koveti
Parent: b7142471

0 script/script-04-preprocess_calmorph_output/script-07-create_plate_map_from_merged_oeretta_layout.R
0 script/script-04-preprocess_calmorph_output/script-08-ploidy.R
0 script/script-04-preprocess_calmorph_output/script-09-big_tbl.R
0 script/script-04-preprocess_calmorph_output/tmp.R
0 script/script-05-GLMM-normalisation/script_03_01-create_joblist.R

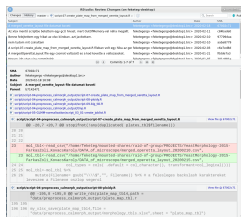
0 script/script-04-preprocess_calmorph_output/script-07-create_plate_map_from_merged_oeretta_layout.R View file @ 67002c71

```
@@ -20,7 +20,7 @@ stopifnot(!any(duplicated( plates_tbl$filename)))
20 20
21 21
22 22
23 mol_tbl<-read_csv("/home/feketeg/mounted-shares/raid-of-group/PROJECTS/YeastMorphology-2015-
FarkasZoli_KovacsKaroly/DATA_of_microscope/merged_oeretta_layout_20200210.csv",
23 mol_tbl<-read_csv("/home/feketeg/mounted-shares/raid-of-group/PROJECTS/YeastMorphology-2015-
FarkasZoli_KovacsKaroly/DATA_of_microscope/merged_oeretta_layout_20200215.csv",
24 24
25 25 col_types = cols( .default = col_character(), transformant=col_logical()))
26 26
27 27 mol_tbl<-mol_tbl %>%
mutate(Filename= gsub("\\\\", "", Filename)) %>% # a felesleges backslash karaktereket
leveszem a Fileneve oszlop vegerol
```

0 script/script-04-preprocess_calmorph_output/script-08-ploidy.R View file @ 67002c71

```
@@ -195,8 +195,8 @@ write_rds(plate_map_tbl4,path =
"data/preprocess_calmorph_output/plate_map_tbl.r
195 195
196 196 my_xlsx_save(plate_map_tbl4,file =
"data/preprocess_calmorph_output/morphology_tbls.xlsx",sheet = "plate_map_tbl")
197 197
```

Let's see how to use it- History



- each row is a commit with
 - date
 - author
 - comment
 - commit ID
- list of files modified in the selected commit
- diffs : for each file it shows what is modified

Principles

- If you want to roll back
 - You have to commit first
 - git will replace the actual files with the old ones
 - You can not rollback only one file.
- You can go back to an old version and then return to the latest version
- If you want to go somewhere you have to tell the ID of the version
- the last commit called **HEAD**
- the 'go to' command is **checkout**

Example

```
git checkout 67002c71
```

```
git checkout HEAD
```

Principles

bad news

sometimes you need to type commands

GitHub

GitHub

GitHub is a webserver where you can save

- GIT saves things to the local repository on your machine
- GIT can upload everything to a remote git-server
- GitHub is one of the git servers

GitHub

GitHub is a webserver where you can save

- GIT saves things to the local repository on your machine
 - local repository needs to exist
- GIT can upload everything to a remote git-server
 - does not upload the actual files.
 - uploads the repository: all versions
- GitHub is one of the git servers
 - Bitbucket
 - Gitlab
 - We can have our own git server

GitHub

GitHub is a webserver where you can save

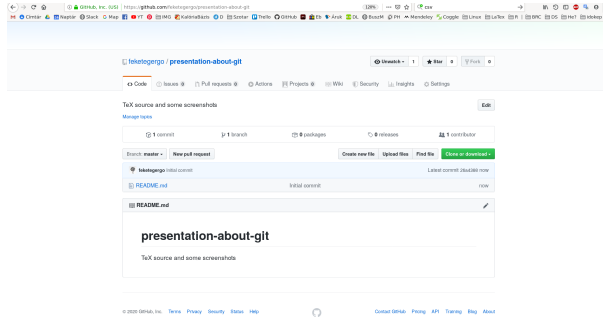
- GIT saves things to the local repository on your machine
 - local repository needs to exist
- GIT can upload everything to a remote git-server
 - does not upload the actual files.
 - uploads the repository: all versions
- GitHub is one of the git servers
 - Bitbucket
 - Gitlab
 - We can have our own git server

GitHub

GitHub is a webserver where you can save

- GIT saves things to the local repository on your machine
 - local repository needs to exist
- GIT can upload everything to a remote git-server
 - does not upload the actual files.
 - uploads the repository: all versions
- GitHub is one of the git servers
 - Bitbucket
 - Gitlab
 - We can have our own git server

- basic git servers just stores the repository
- GitHub provides additional Web interface



Start a GitHub Project

- go to github.com , register a user
- create a new project. Tick the 'initialised' checkbox.
- copy paste the url of the project
- start a terminal, go to the parent folder.
- use 'git clone *<url>*' command
- now you have an initialised local repository in the folder
- the **clone** command automaticly connected it to the GitHub repo.
- you can commit files.
- if you press the 'push' button or give the 'git push' command, then everything will be uploaded.
- If another user modified the files on the server the 'git pull' command download it

Start a GitHub Project

- go to github.com , register a user
- create a new project. Tick the 'initialised' checkbox.
- copy paste the url of the project
- start a terminal, go to the parent folder.
- use 'git clone <url>' command
- now you have an initialised local repository in the folder
- the **clone** command automaticly connected it to the GitHub repo.
- you can commit files.
- if you press the 'push' button or give the 'git push' command, then everything will be uploaded.
- If another user modified the files on the server the 'git pull' command download it

Tricky things start here

- If you try to upload a file, what is modified by another user. . . .
- It is called 'conflict'
- The operation 'merge' can fix the problem
- normally git merge it automaticly
- pull first then push

Tricky things start here

- If you try to upload a file, what is modified by another user. . . .
- It is called 'conflict'
- The operation 'merge' can fix the problem
- normally git merge it automaticly
- pull first then push

Git Data Transport Commands

<http://osteele.com>