

Laboratorijas darbs versiju kontrolē ar git

Lab in version control with git

Programminženierija, LU Datorikas fakultāte
Software engineering, University of Latvia, Faculty of Computing
2023

Laboratorijas darba vispārējie noteikumi //

General rules for the lab assignment

- Balstoties uz aprakstu izveidot git repozitoriju:
 - github vai alternatīvā git serverī
 - dokumentēt veiktos soļus
 - ievērot failu nosaukumu nomenklatūru – failu nosaukumiem jā sākas ar studenta apliecības numuru, piemēram, jz93015-main.py
- Iesniedzt atskaiti par padarīto (sk. tālākos slaidus)
- Following the description, create a git repository:
 - in github or an alternative server
 - document steps accomplished
 - obey file naming conventions – names should start with student' id, e.g., jz93015-main.py
- Submit report on the accomplished work (see next slides)

Laboratorijas darba atskaites kopsavilkums //

Summary of report on the laboratory work

1. Ekrāna kopija no github (1 attēls)
2. Darba procesa vizualizācija, piemēram, ar revision graph (4 attēli)

(Visi attēlu faili var tikt apvienoti vienā pdf failā.)

3. Lokālā repozitorija beigu stāvokļa kopija zip formātā (1 zip fails)
4. Failus iesniegt e-studijās

1. Screenshot of github (1 image)
2. Visualization of the process, e.g., with revision graph (4 images)

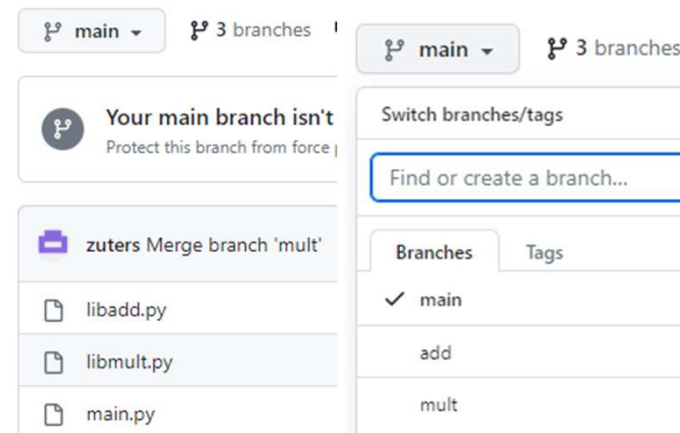
(All image files can be combined to a single pdf file.)

3. Local copy of the final status of the repository
4. Submit to e-studies

1/4. Ekrāna kopija no github beigu stāvoklī //

1/4. Screenshot of github in the final status

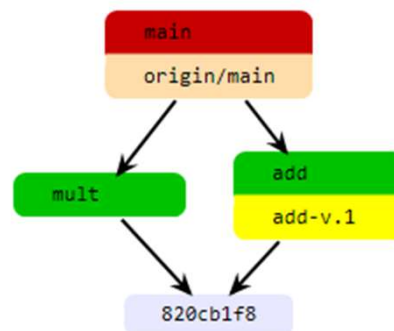
- Ekrāna kopija no github repozitorija beigu stāvoklī, kurā redzams konts, repozitorija nosaukums, visu zaru nosaukumi un visu failu nosaukumi (1 attēls)
- Screenshot of github in the final status of the repository including account name, repository name, names of all branches, all file names (1 image)



2/4. Darba procesa vizualizācija //

2/4. Visualization of the process

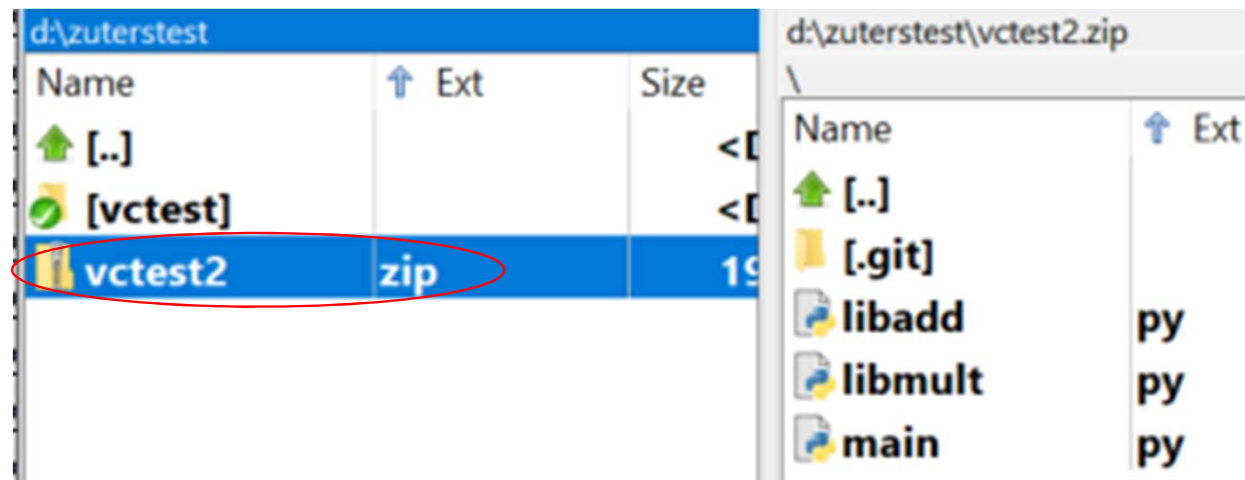
- Darba procesa vizualizācija 4 starpstāvokļiem, piemēram, ar revision graph (4 attēli):
 - sākotnējā versija
 - add atzars izveidots
 - mult atzars izveidots
 - beigu versija
- Visualization of the laboratory work process for the 4 intermediate states (4 images):
 - initial version
 - add branch created
 - mult branch created
 - final status



3/4. Lokālā repozitorija kopija zip formātā //

3/4. Zipped local copy of the repository

- Lokālā repozitorija beigu stāvokļa kopija zip formātā (1 zip fails)
- Local copy of the final status of the repository (1 zip file)



4/4. Failu iesniegšana e-studijās //

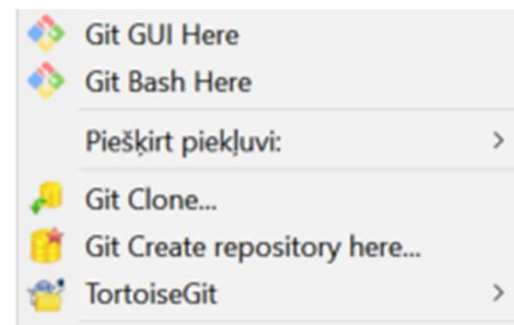
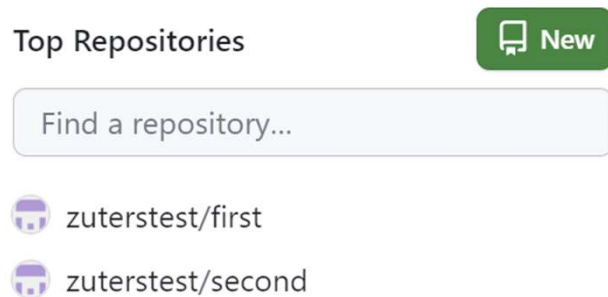
4/4. Submission of the files in e-studies

- Iesniegt 5 attēlus (iespējams kā vienu pdf failu) un vienu zip failu
- Atļautie paplašinājumi *.jpg, *.pdf un *.zip
- Submit 5 images (possibly as one pdf file) and one zip file
- Accepted extensions *.jpg, *.pdf and *.zip

Sākotnējie nosacījumi laboratorijas darbam //

Prerequisites for the lab work

- Izveidot kontu github.com, piemēram, zuterstest
- (Windows) instalēt TortoiseGit:
 - <https://tortoisegit.org/support/faq/#prerequisites>
 - <https://tortoisegit.org/download/>
- Sign up to github.com, e.g., zuterstest
- (Windows) install TortoiseGit:
 - <https://tortoisegit.org/support/faq/#prerequisites>
 - <https://tortoisegit.org/download/>



Izveidot jaunu repozitoriju github //

Create new repository in github

Start a new repository

A repository contains all of your project's files, revision his

zuterstest /

vctest



Public

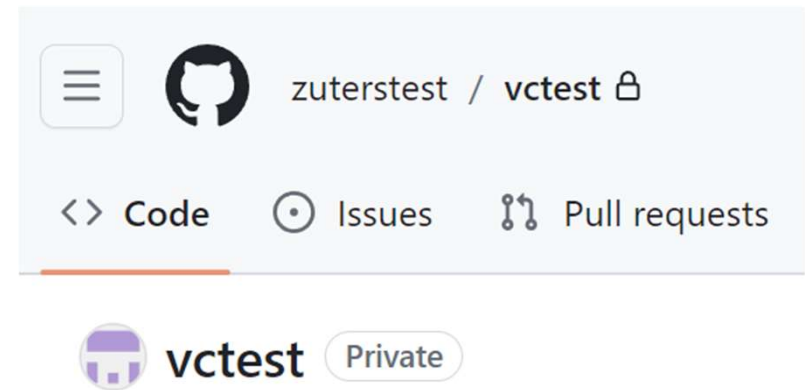
Anyone on the internet can see this repository



Private

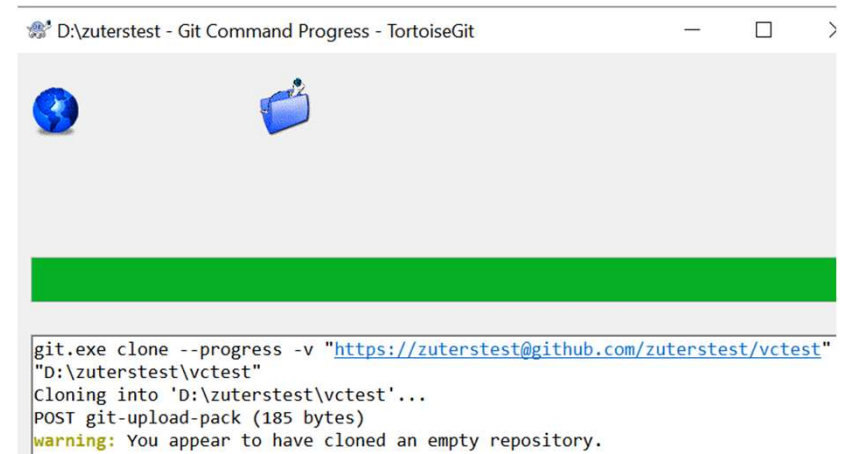
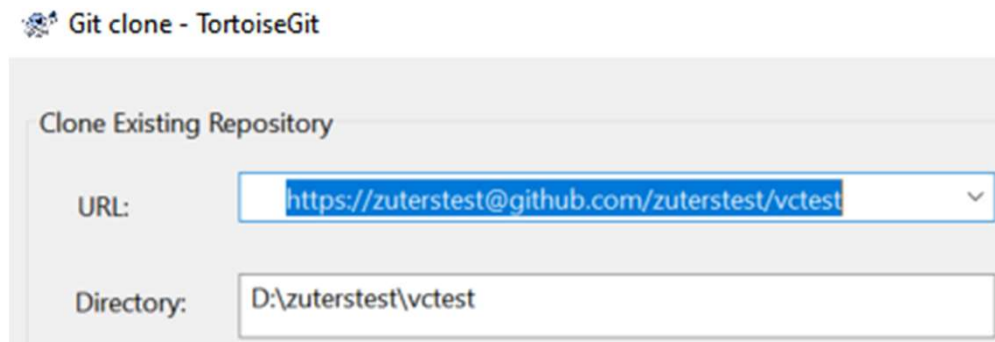
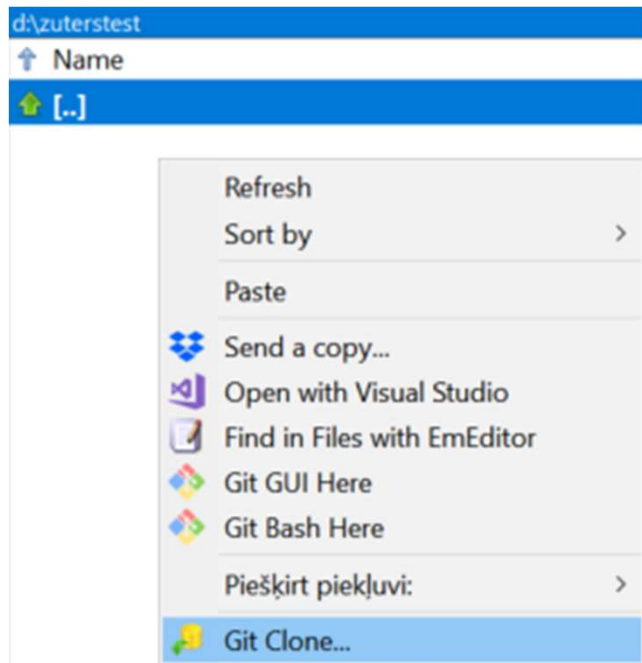
You choose who can see and commit to this repository

Create a new repository



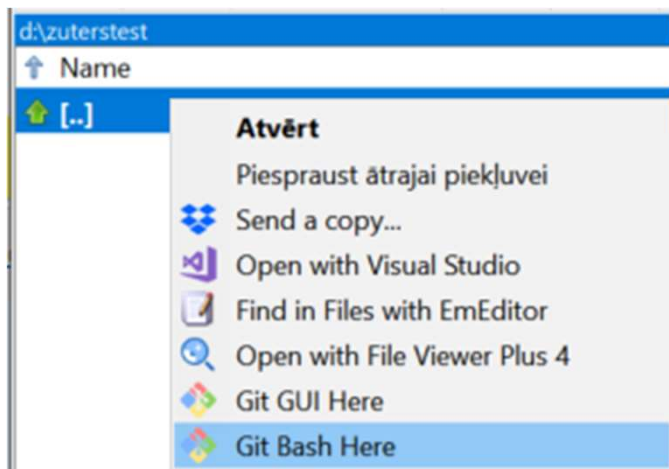
Klonēt repozitoriju failu sistēmā //

Clone the repository into local file system



Klonēt repozitoriju failu sistēmā v.2 //

Clone the repository into local file system v.2



git clone [https://\[username\]@github.com/\[username\]/\[repository-name\].git](https://[username]@github.com/[username]/[repository-name].git)

git clone [https://git@github.com/\[username\]/\[repository-name\].git](https://git@github.com/[username]/[repository-name].git)

git clone <https://zuterstest@github.com/zuterstest/vctest.git>

MINGW64:/d/zuterstest

janisz@jzpc2018 MINGW64 /d/zuterstest

```
$ git clone https://zuterstest@github.com/zuterstest/vctest.git
```

MINGW64:/d/zuterstest

janisz@jzpc2018 MINGW64 /d/zuterstest

```
$ git clone https://zuterstest@github.com/zuterstest/vctest.git
```

Cloning into 'vctest'...

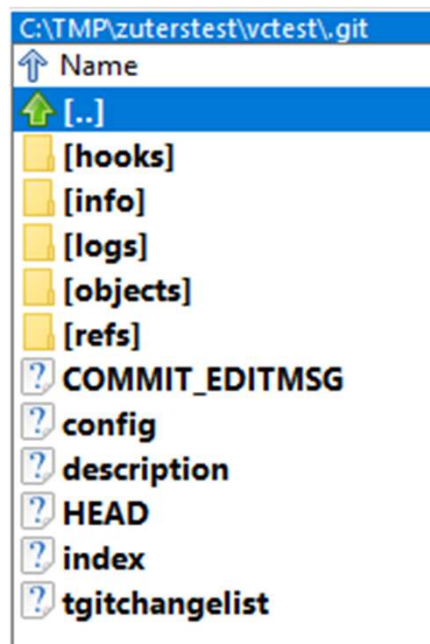
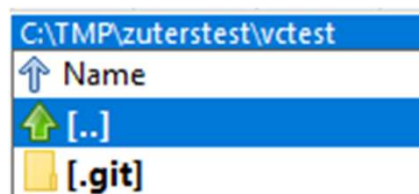
warning: You appear to have cloned an empty repository.

janisz@jzpc2018 MINGW64 /d/zuterstest

```
$
```

Klonēšana // Cloning

- Tiek izveidota jauna direktorija ar repozitorija lokālo kopiju
- Jaunajā repozitorijā atrodas:
 - aktuālie faili (ja tādi ir)
 - direktorija .git, kurā ir visa versiju kontroles informācija par doto repozitoriju un saite uz serveri



- A new directory with the local copy of the repo is created
- In the new repo you can find:
 - actual files (if any)
 - directory .git to hold the whole versioning information and link to server

Iekopēt failus tukšajā repozitorijā //
Copy the files into the empty repository

```
C:\TMP\zuterstest\yctest
Name                               Ext
[.]
[.git]
libadd                             py
libmult                             py
main                               py
```

libadd.py libmult.py main.py

plus

```
1 def plus(a,b):
2     return a+b
```

libadd.py libmult.py main.py

mult

```
1 def mult(a,b):
2     return a*b
```

libadd.py libmult.py main.py

```
1 # This is a simple calculation program
2 # created to demonstrate version control.
3 from libadd import *
4 from libmult import *
5 a = 13
6 b = 5
7 print(plus(a,b))
8 print(mult(a,b))
```

Console:
18
65

Pievienot failus repozitorijam, veikt commit //

Add files to repo and perform initial commit

TortoiseGit//add

The image displays three sequential screenshots of the TortoiseGit application interface, illustrating the process of adding files to a repository and performing an initial commit.

Left Screenshot: 'Add' Dialog
The dialog shows a list of 'Not Versioned Files' with the following columns: Path, Extension, and a checkbox. The files listed are libadd.py, libmult.py, and main.py, all with a .py extension. At the bottom, there are checkboxes for 'Select/deselect all' and 'Include ignored files', and an 'OK' button.

Middle Screenshot: 'Add Finished!' Dialog
This dialog shows the results of the 'Add' operation. It has a table with columns 'Action' and 'Path'. The actions listed are 'Command Add', 'Added libadd.py', 'Added libmult.py', 'Added main.py', and 'Finished! Success (78 ms @ 06.10.2023 13:55:...)'. A 'Commit...' button is visible at the bottom right.

Right Screenshot: 'Commit' Dialog
The 'Commit' dialog shows the commit message 'Initial commit' and options for 'Commit to: main' and 'new branch'. It also includes checkboxes for 'Amend Last Commit', 'Set author date', and 'Set author'. Below this, there is a section for 'Changes made (F5: refresh, double-click on file for diff):' with a table showing the status of the files. The table has columns: Path, Extension, Status, Lines added, and Lines removed. The files listed are libadd.py, libmult.py, and main.py, all with a .py extension, and their status is 'Added'. At the bottom, there are checkboxes for 'Staging support (EXPERIMENTAL)', 'Show Unversioned Files', and 'Do not autoselect submodules', and a 'Commit' button.

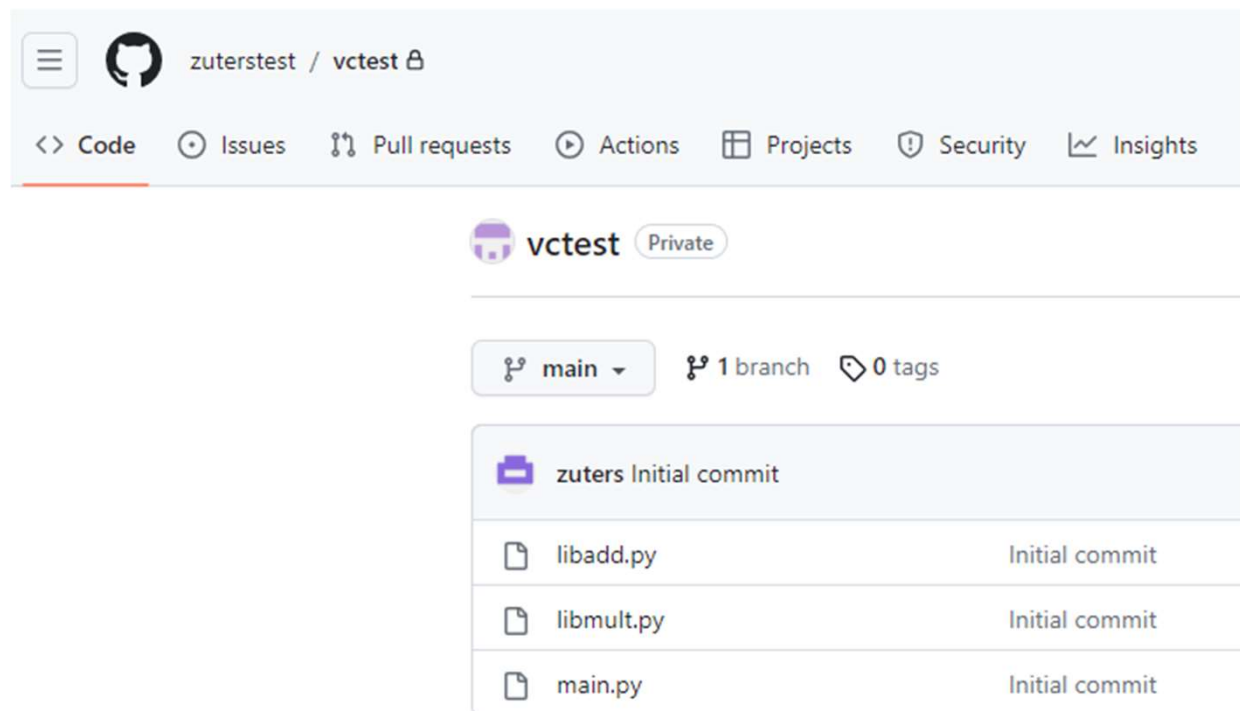
Tagad visi 3 faili ir pievienoti repozitorija lokālajai kopijai //

All the three files are added to the local copy of the repository

Izmaiņu ielādēšana serverī ar push //

Pushing changes to the server

Git Sync – Push

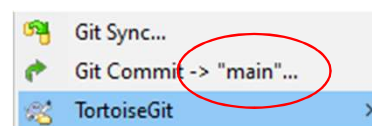


The screenshot shows the GitHub interface for a repository named 'vctest' under the user 'zuterstest'. The repository is marked as 'Private'. The navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Security, and Insights. Below the navigation bar, the repository name 'vctest' is displayed with a 'Private' label. A dropdown menu shows the current branch as 'main', with '1 branch' and '0 tags' indicated. Below this, a table lists the files in the repository, all from the 'Initial commit' by 'zuters'.

File	Commit
libadd.py	Initial commit
libmult.py	Initial commit
main.py	Initial commit

Galvenais zars main // Branch main

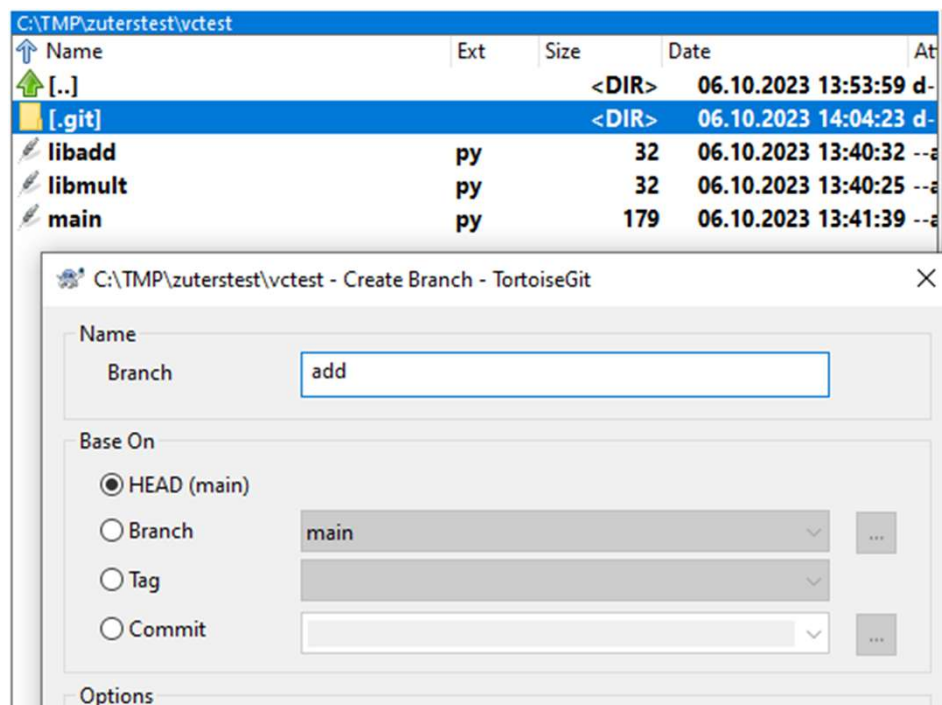
- Šobrīd repozitorijā atrodas tikai (noklusētais) galvenais versiju zars main (kādreiz sauca master) un tikai sākotējā versija
- no lokālās kopijas skatu punkta:
 - main – lokālais zars main
 - origin/main – zars main uz servera
- By now, there is only one (default) versioning branch in the repository – called main (previously – master) and just the initial version
- From the local copy perspective:
 - main – the local main branch
 - origin main – the branch main on server



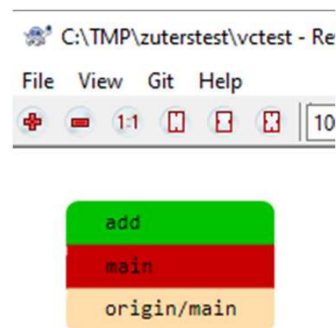
Jauna zara 'add' izveidošana //

Creating a new branch 'add'

TortoiseGit//Create Branch



TortoiseGit//Revision Graph

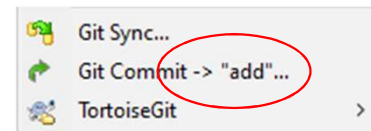
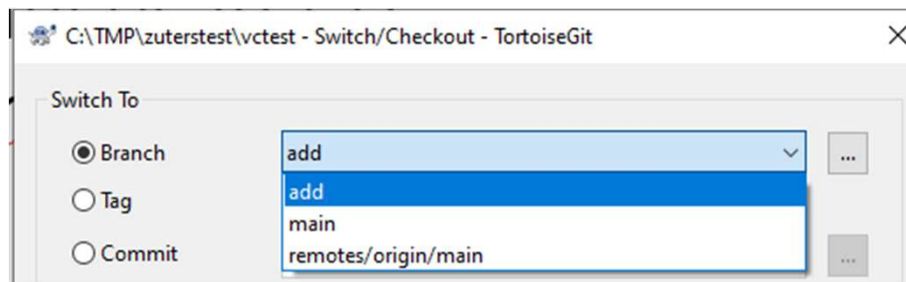


- Izveidots jauns zars, kurš ir identisks main, aktīvais zars ir joprojām ir main (red)
- A new branch is created, and it is still identical to main, the active branch is main (red)

add zara modificēšana //

Mofifying branch add

TortoiseGit//Switch/Checkout – «add»



```
libadd.py  libmult.py  main.py
plusx
1 def plus(a,b):
2     return a+b
3 def plusx(a,b,c):
4     return a+b+c
```

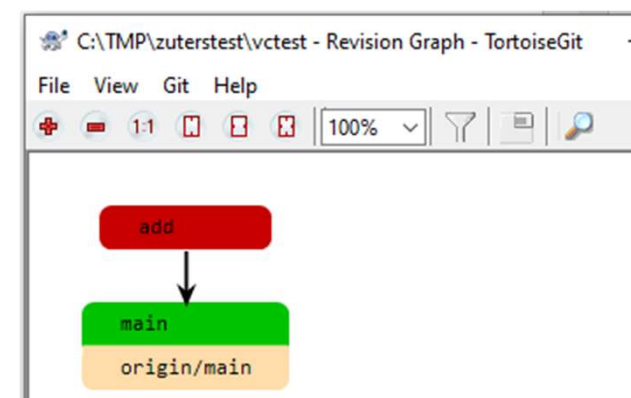
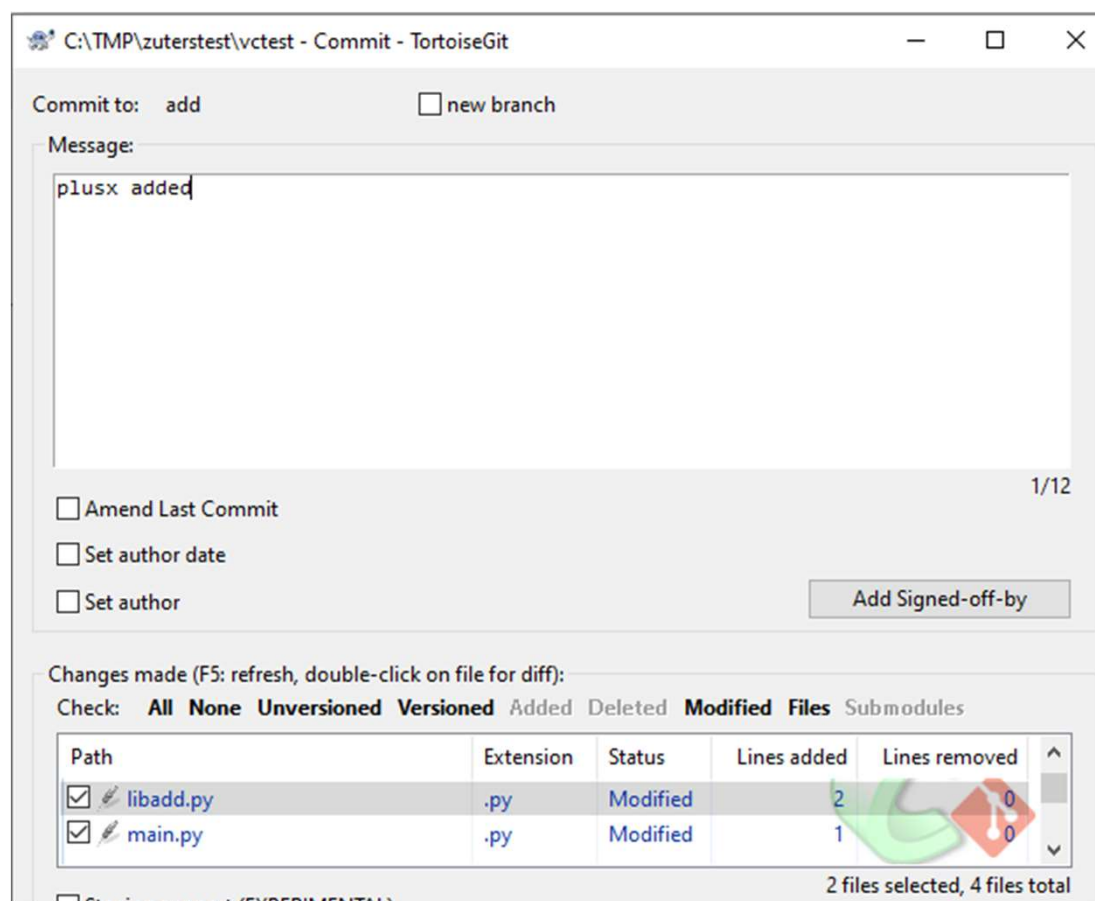
```
libadd.py  libmult.py  main.py
1 # This is a simple calculation program
2 # created to demonstrate version control.
3 from libadd import *
4 from libmult import *
5 a = 13
6 b = 5
7 print(plus(a,b))
8 print(plusx(a,b,2))
9 print(mult(a,b))
```

Console:

18
20
65

add zara commit // commit on add branch

git commit

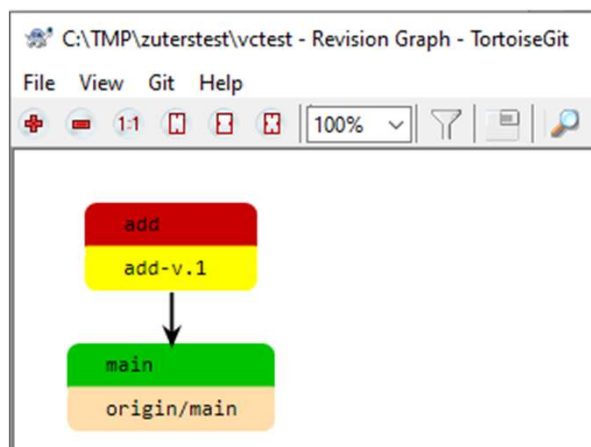


- Aktīvais zars ir add (sarkans) un tas ir atšķirīgs no main
- Active branch is add (red) and it is different than main

Taga (markiera) pievienošana //

Adding a tag

TortoiseGit//Create Tag – «add-v.1»



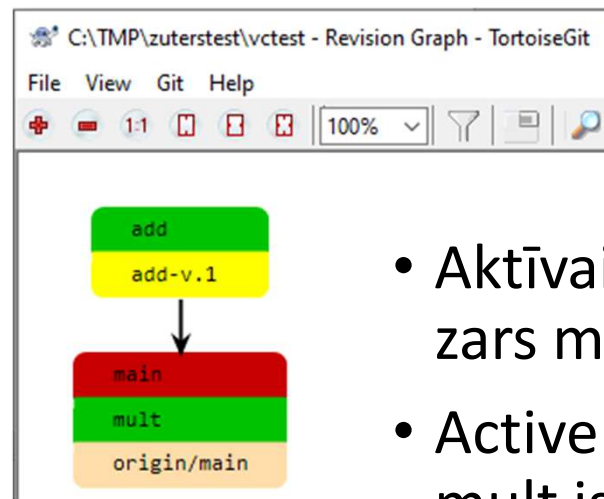
Otra zara 'mult' izveidošana //

Creating a new branch 'mult'

TortoiseGit//Switch/Checkout – «main»

TortoiseGit//Create Branch – «mult»

TortoiseGit//Revision Graph



- Aktīvais zars ir main, bet zars mult ir tam identisks
- Active branch is main, but mult is equal to it

mult zara modificēšana // Modifying branch mult

TortoiseGit//Switch/Checkout – «mult»

```
libadd.py libmult.py main.py  
multx  
1 def mult(a,b):  
2     return a*b  
3 def multx(a,b,c):  
4     return a*b*c
```

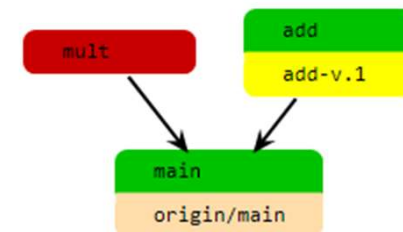
```
libadd.py libmult.py main.py  
1 # This is a simple calculation program  
2 # created to demonstrate version control.  
3 from libadd import *  
4 from libmult import *  
5 a = 13  
6 b = 5  
7 print(plus(a,b))  
8 print(mult(a,b))  
9 print(multx(a,b,2))
```

Console:

```
18  
65  
130
```

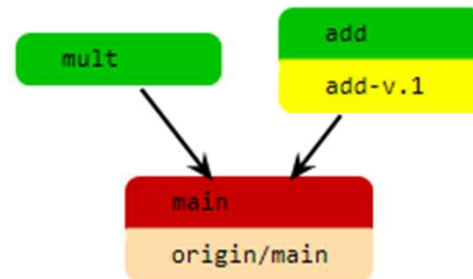
git commit -m «add multx»

TortoiseGit//Revision Graph

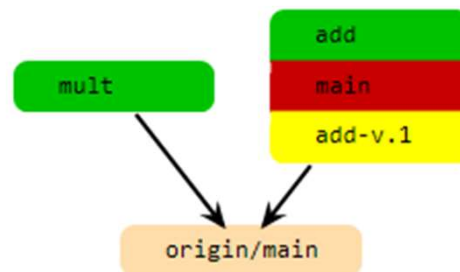


zara add pievienošana main zaram //
merging main branch from add

TortoiseGit//Switch/Checkout – «main»

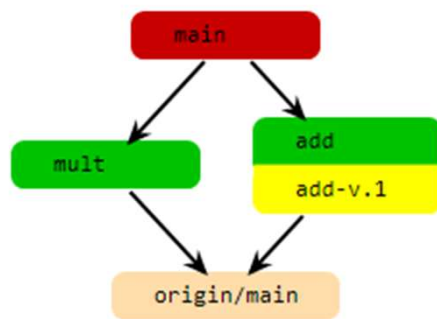


TortoiseGit//merge – from «add»



zara mult pievienošana main zaram // merging main branch from mult

TortoiseGit//merge – from «mult»



```
libadd.py  libmult.py  main.py
< > plusx v
1  def plus(a,b):
2      return a+b
3  def plusx(a,b,c):
4      return a+b+c
```

```
libadd.py  libmult.py  main.py
< > multx v
1  def mult(a,b):
2      return a*b
3  def multx(a,b,c):
4      return a*b*c
```

```
libadd.py  libmult.py  main.py
< >
1  # This is a simple calculation program
2  # created to demonstrate version control.
3  from libadd import *
4  from libmult import *
5  a = 13
6  b = 5
7  print(plus(a,b))
8  print(plusx(a,b,2))
9  print(mult(a,b))
10 print(multx(a,b,2))
```

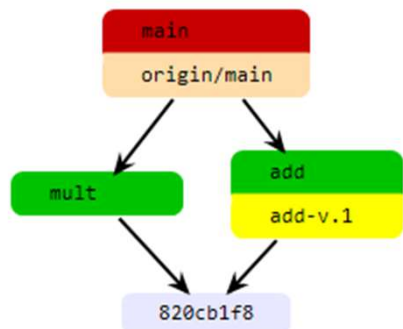
Console:

```
18
20
65
130
```


Izmaiņu ielādēšana serverī ar push //

Pushing changes to the server

Git Sync – Push



TortoiseGit//Switch/Checkout – «add»

Git Sync – Push

TortoiseGit//Switch/Checkout – «mult»

Git Sync – Push

