# Lab: Version control systems

Problems for exercises and homework for the [“Programming Fundamentals” course @ SoftUni](https://softuni.bg/courses/programming-fundamentals).

# Create your own GitHub developer profile

## Create profile

Register for a free **developer account at GitHub**: <http://github.com>. Submit your developer profile's URL as output of this homework.



# Create a repo, make conflicts and resolve them.

## Create a GitHub Repository

* New repository form: <https://github.com/new>.
* Choose a name for the repo, e.g. "first-repo". Make sure to "**Initialize this repository with a README**".
* Upload simple "**test.txt**" file with sample content inside.



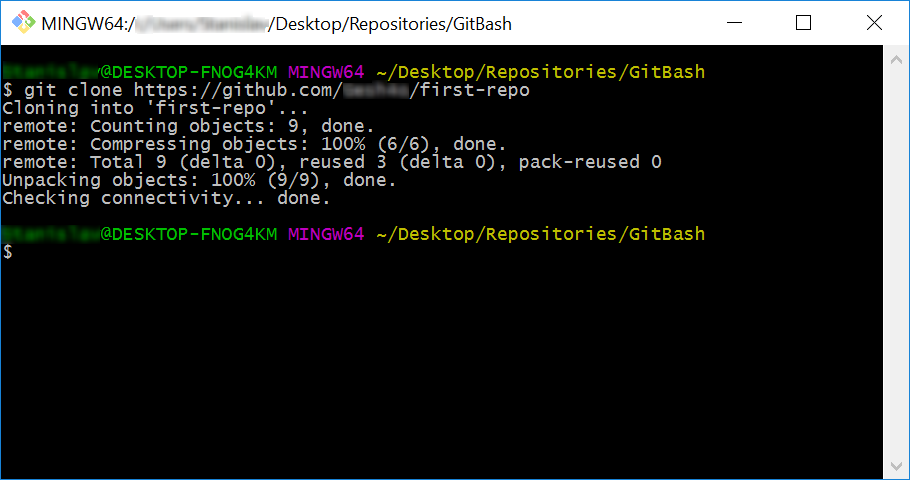
## Clone that repository on two different places on your personal device.

* Use Git **clone** for cloning with **TortoiseGit**.
  + Go in the desired directory, right click on blank space anywhere in the folder and copy the link of your repository.
* The result should be something like this:
* Use *"***git clone***"* command for cloning with **GitBash**.
  + Go to the desired **directory**, right click on blank space anywhere in the folder, select "**Git Bash here**" and type "git clone" command followed by the link of your repository.





* + The result should be something like this:



## Update content in both directories differently.

* On your **TortoiseGit** clone open the **test.txt** file and add line: ***"*Update with Tortoise…*"***
* On your **GitBash** clone open the **test.txt** file and add line: ***"*Updating with Bash…*"***

## Upload your changes from TortoiseGit clone.

* You can use TortoiseGit's *"***Git Commit…***"*:





## Try to update your Bash clone.

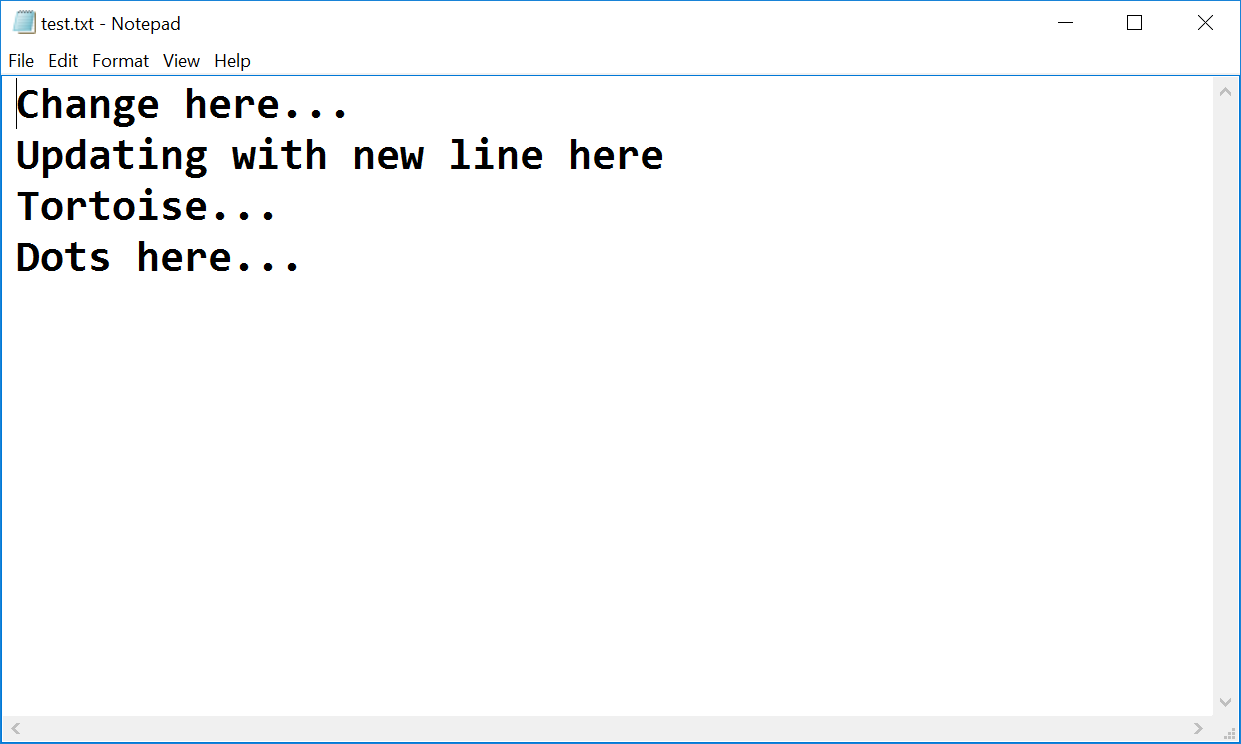
* Open your Git clone directory and open **GitBash** console. Run the following commands:
  + Add all modified files to **staging** area
    - "**git add .**"
  + **Commit** your changes and a give commit message.
    - **''git commit -m "Update test.txt."*"***
  + **Update** your local repository
    - "**git pull**"

## Now you have merge conflict which you have to resolve.

* + **Open** the **test.txt** file in your **GitBash** clone, it should look like this: 
  + Remove the HEAD, ======, <<<<<<, >>>>>>> symbols and save the file.
* Now that you have resolved the **conflict - stage** the modified file, **commit** again and **sync** with the remote repository.

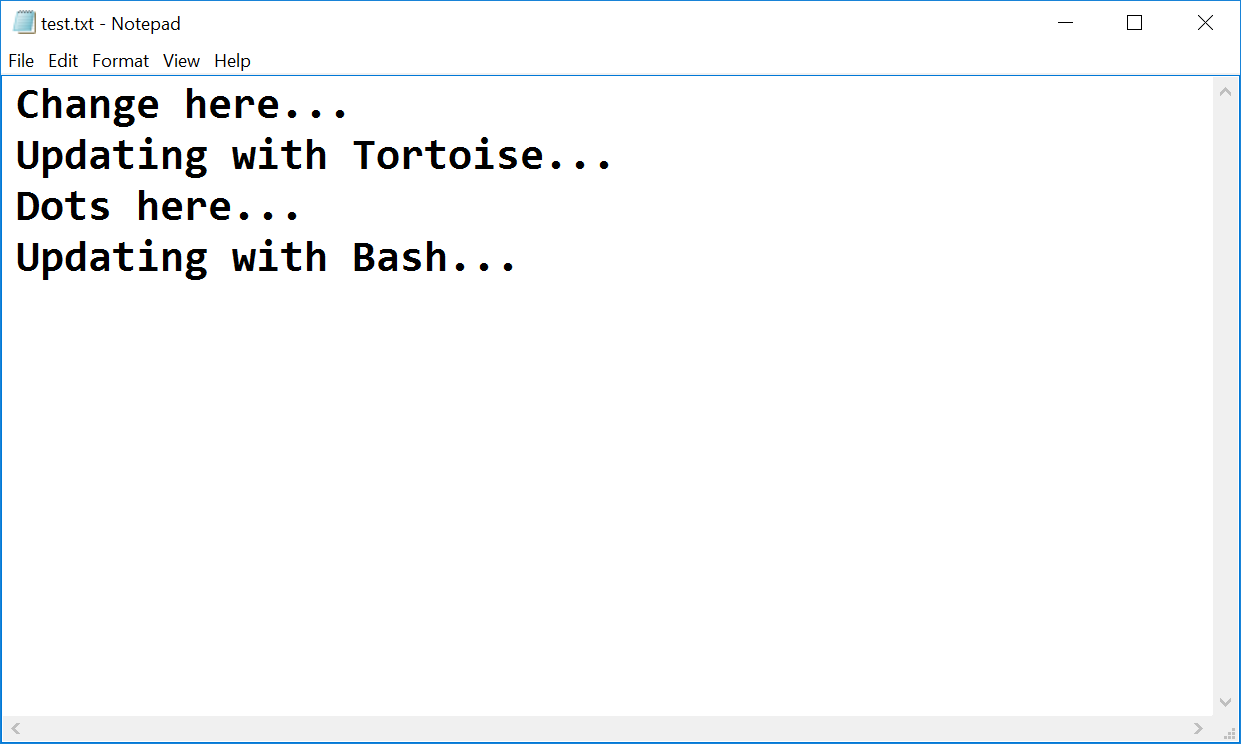
## You have updated the content of your remote repository, now try to update your TortoiseGit clone.

* Make additional changes to test.txt and **commit** them.

**\*Note** that if you make changes too simple TortoiseGit may **automatically** merge them.

* Now try to **push**. It turns out that we have our **remote** repository **updated** (the merge commit) and we do not have these changes on our **local** repository.



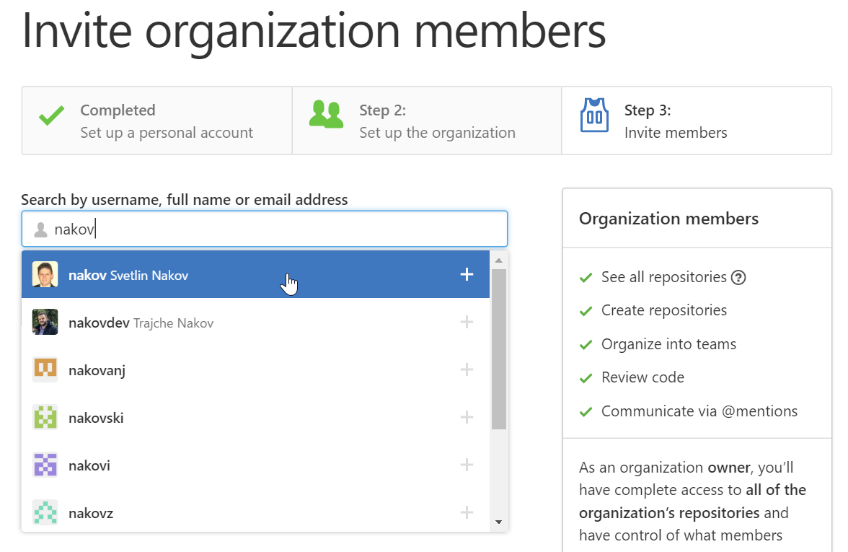
* So we have to **pull** new changes:
* Note that message: "Automatic merge failed; fix conflicts…". We have another conflict and we have to resolve it like we did earlier but small difference:
  + Go on the **test.txt** file. You should **open** the **file** and **remove** the same **symbols** that we have previously removed. Then right click on the file - choose **TortoiseGit** -> **Resolve…** and click it. A dialog window should open. Then you click "*Ok*" in order to try to **resolve** the conflict.
* Now our file is **clean** and we are ready for our final **commit**!

# Teamwork

Work into **teams** of (about) 5 students in class

* Online students work alone or form own teams.
* Each team selects a "**team leader**".

The team leader **creates an organization** in GitHub:

* New organization from: <https://github.com/settings/profile>. Then on the tab: *Organizations*.
* Choose a **unique** name for the organization, e.g. "SoftUniOrg" and add members to it.
* Then create a repository, e.g. "**test-repo**"

## Add a File to GitHub

Team members add a few files:

1. Clone the "**test-repo**" into your computer (if not cloned yet)
2. Create a new file into your working directory
   * Name the new file **<your\_name>.txt**
   * Put some text in it the file, e.g. "*My name is …*"
3. Commit the **new file** to your **local repository**.
4. Sync the changes to **upload your file to the remote repo**.
5. Browse the repo from <https://github.com/user/repo> to check whether your file is successfully uploaded in GitHub.

## Create a Git Conflict & Merge

* All team members create a common file **config.txt**
* Each team member adds some settings in **config.txt**, e.g.
  + *name = Peter*
  + *size = 100*
  + *email = peter@dir.bg*
* Each team member **commits** his local changes.
* Each team member **syncs** his changes.
  + The first member will succeed without **conflicts**.
  + The others will have a **conflict** to be merged.
  + **Resolve** the conflict:
    - **Edit** the merged changes + **commit** and **sync** again.