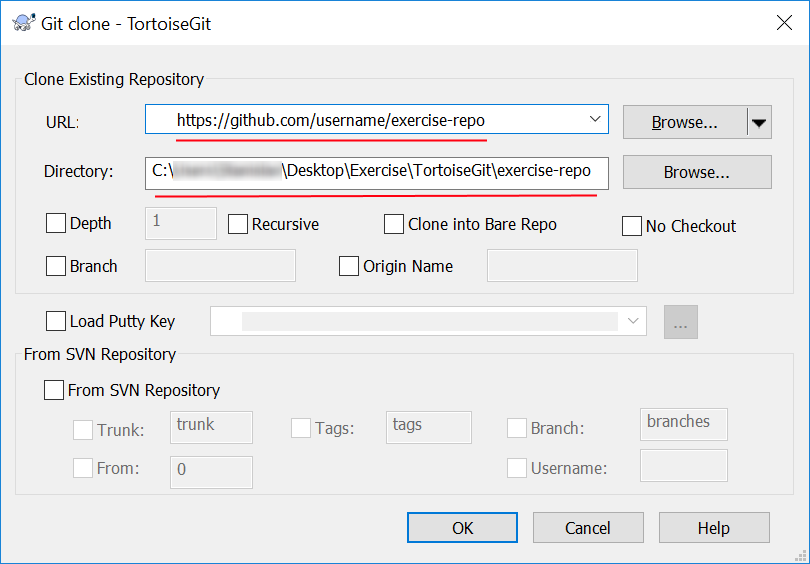
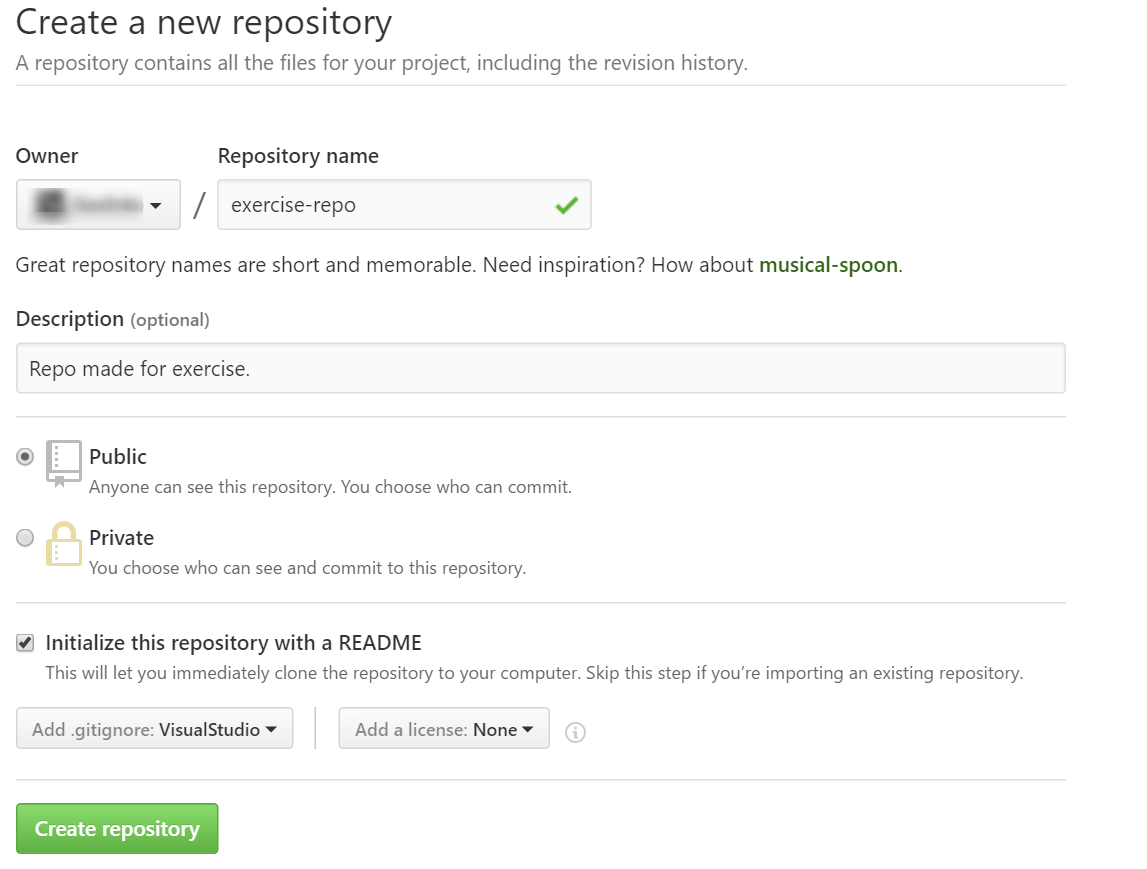
# Exercises: Version Control Systems

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

# TortoiseGit

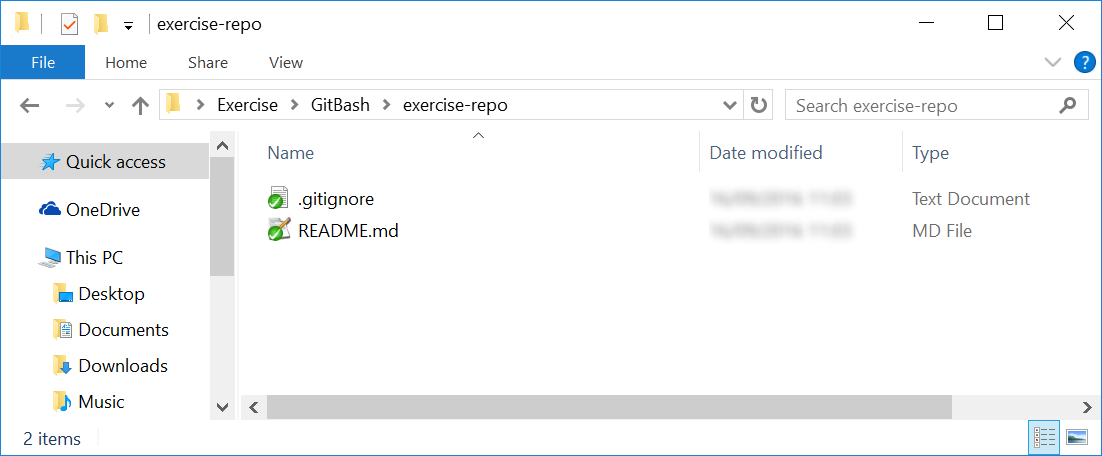
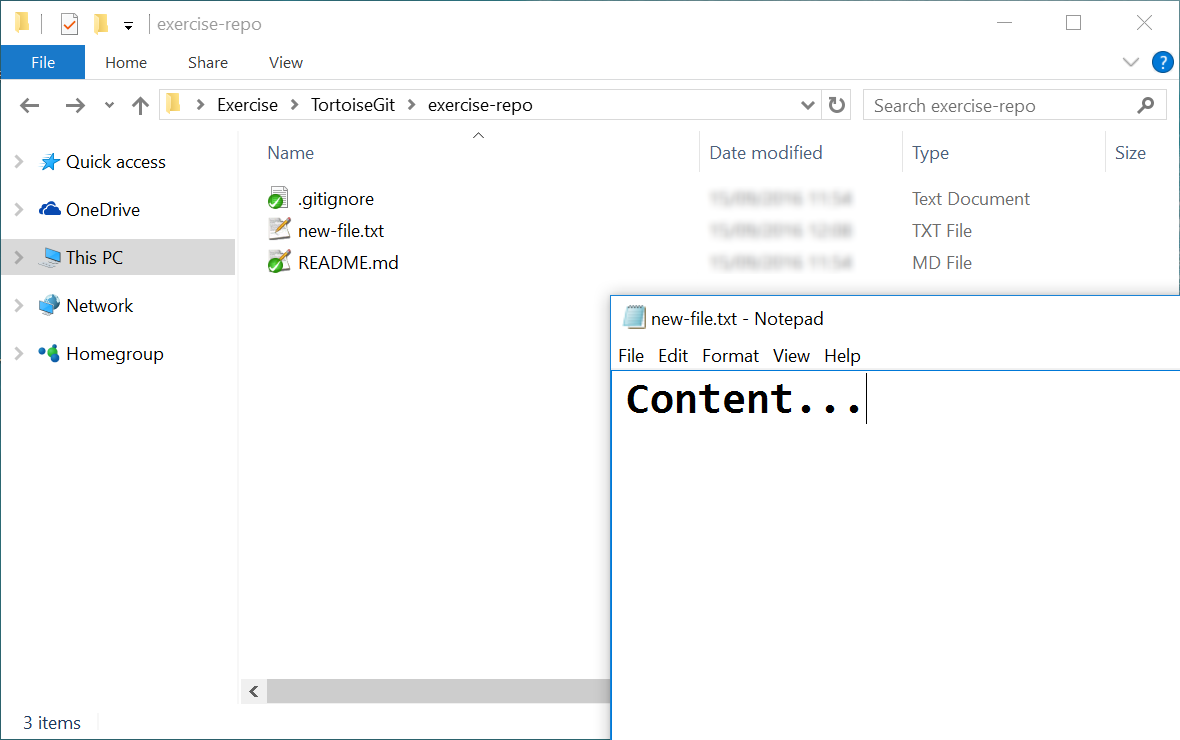
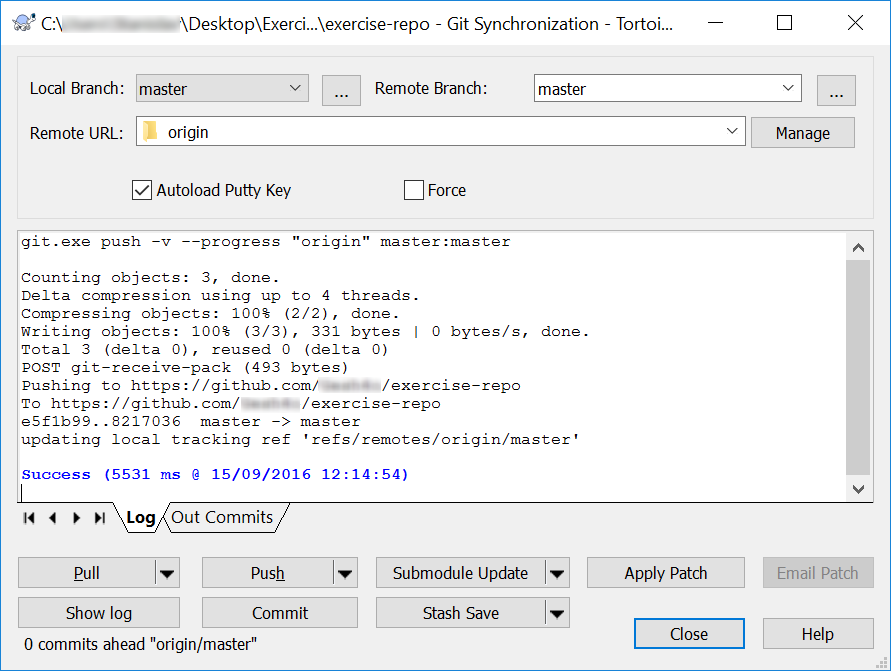
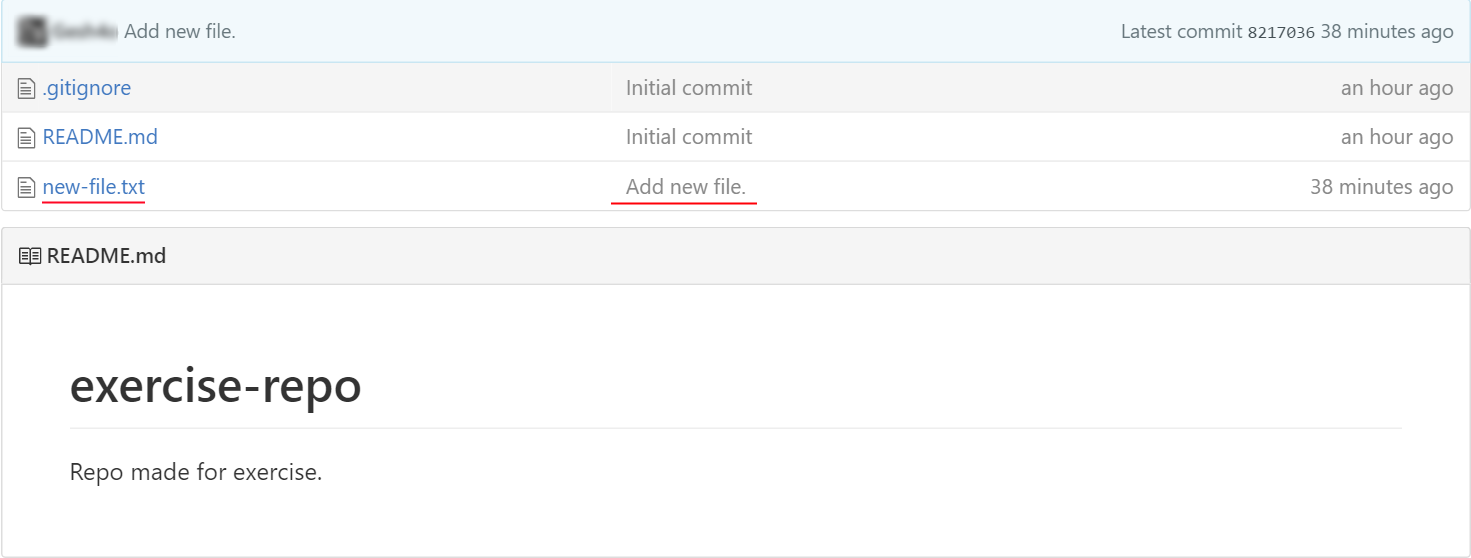
## Upload a Few Projects at GitHub

Create a few **repositories** in your **GitHub** profile and **upload a few of your projects to GitHub**. These could be your **homework exercises** for the last few courses, your **teamwork projects** or any other projects that you might want to share with the developer community. Follow these steps:

1. First **create a remote repository** for your current project.
2. **Clone** it on your device:

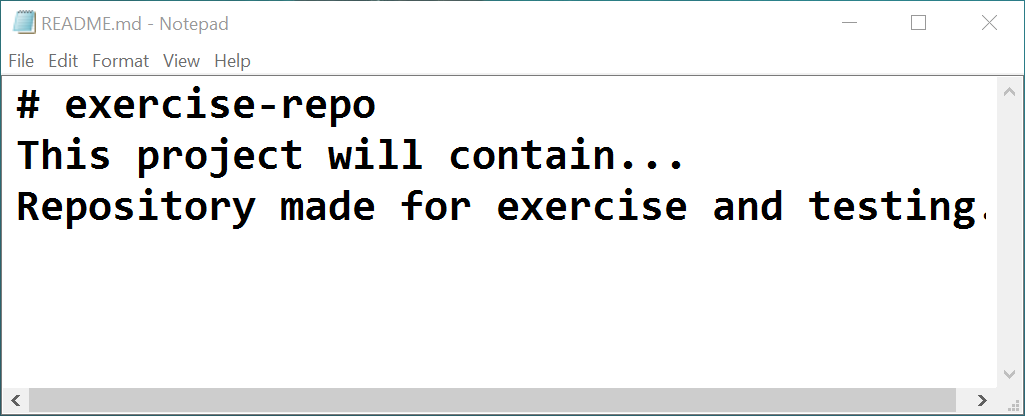
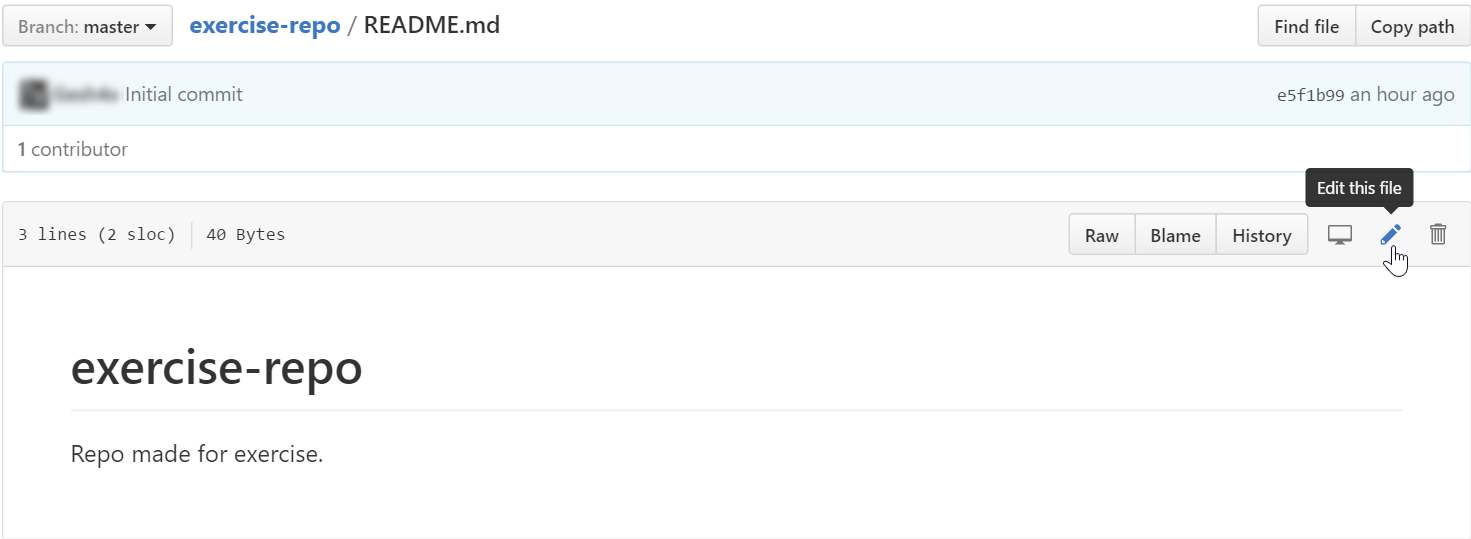
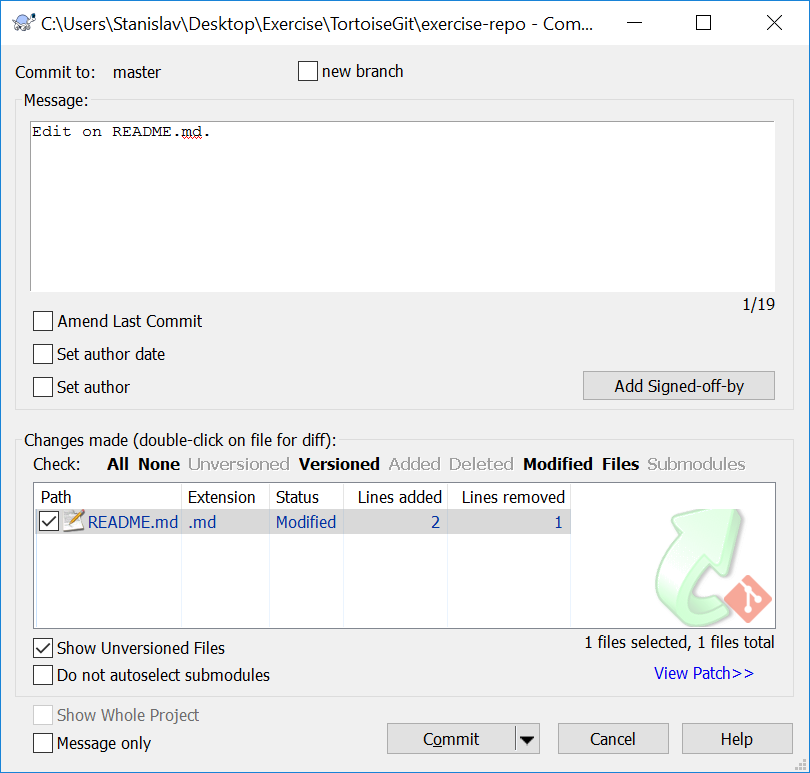
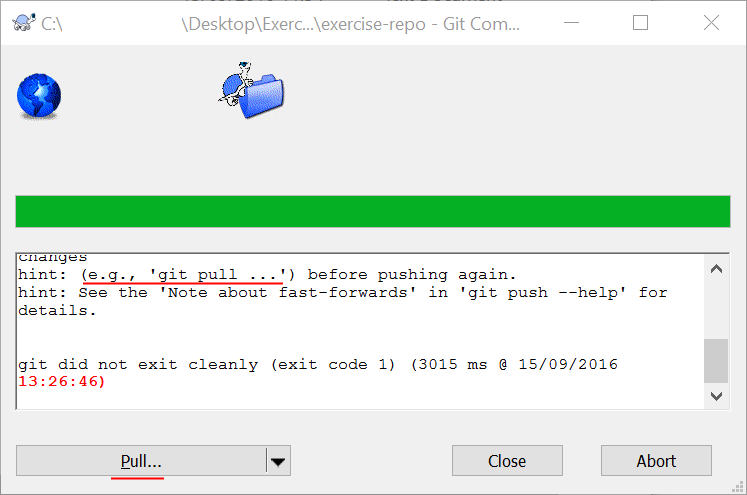
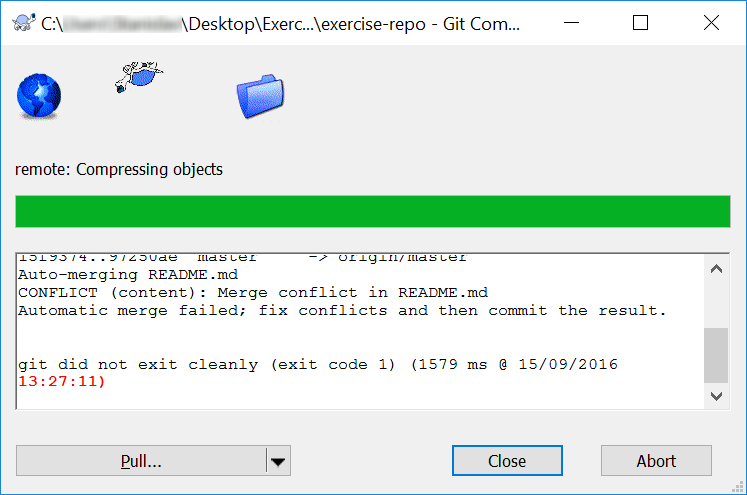
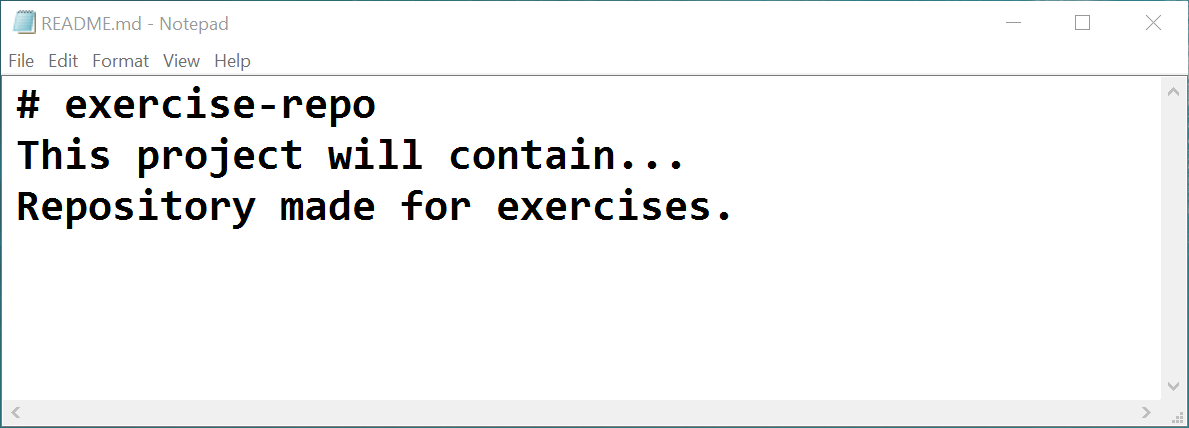
Note that all (not paid) projects you upload at GitHub will be **open-source** and will be accessible for **anyone** in Internet, so be careful about passwords or code which you might **not** want to be **visible** by **someone** else.

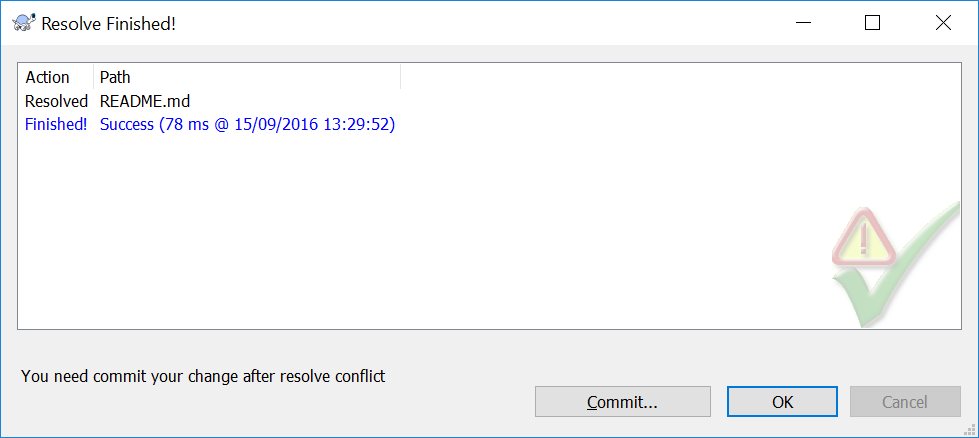
**Clone** some of your GitHub repositories through your **Git client** (e.g. using the **GitBash** or **TortoiseGit**). Make some **changes** locally, then **update** them to GitHub. Check whether the changes are published in your GitHub profile in Internet. Follow these steps:

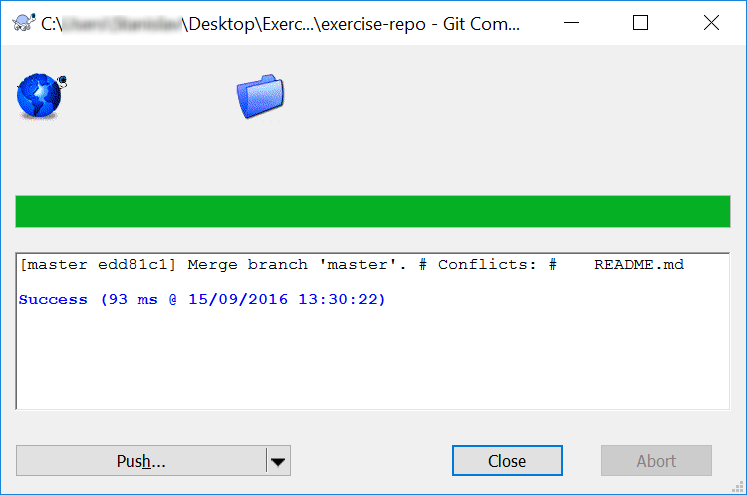
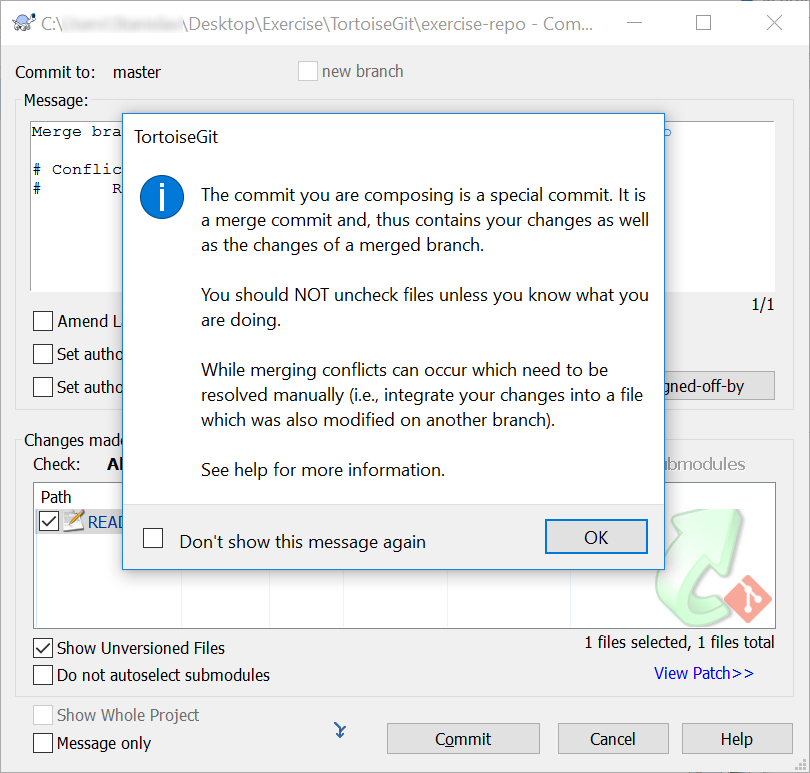
1. **Clone** again the same repository on a different place (this time use **GitBash** -*"***git clone***"*):
2. Return to the previous clone and it in **Windows Explorer**.
3. ****Make some **changes**:
4. **Commit** your local changes to your local repository.
5. **Push** your changes to the remote repository in GitHub:
6. Check whether your changes are online: 

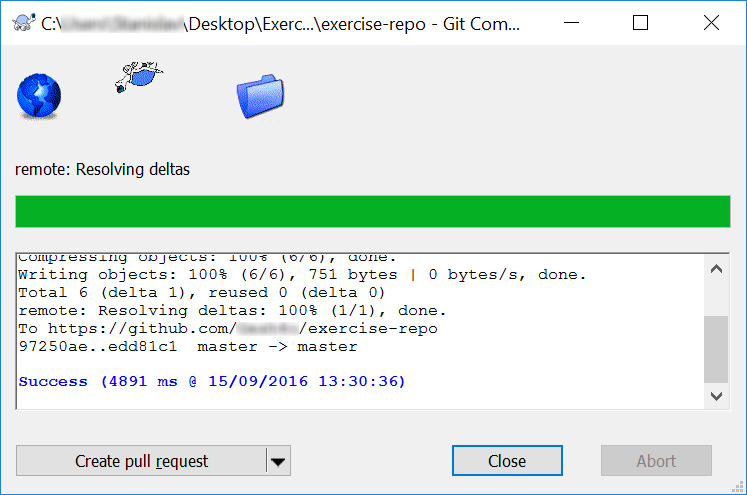
## Make Conflicts and Resolve Them

Create **conflicting changes** and **merge them**. Use the following steps:

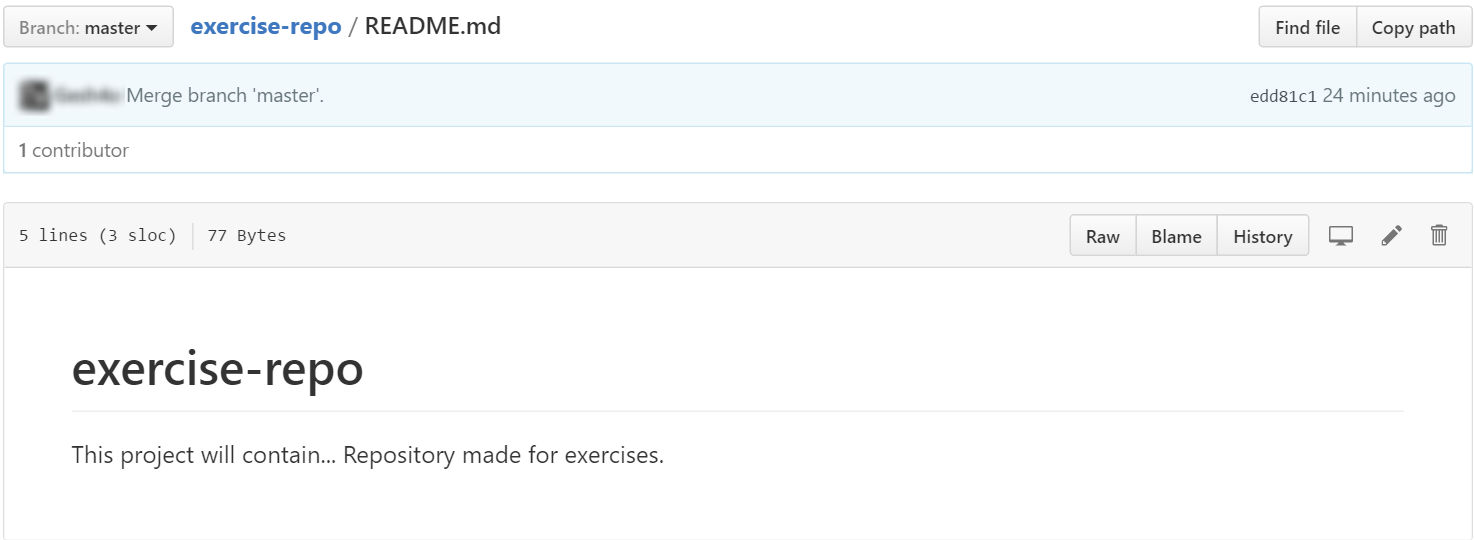
1. ****Make some **changes** in your working directory, e.g. edit the file README.md.
2. **Don’t commit** and **don’t push** your changes yet.
3. Open your GitHub account from your **Web browser** or your **GitBash clone**. Make some changes to the same file, commit and push them:
4. Now **commit** the local changes in **TortoiseGit** clone.
5. Try to **push** the local changes to the **remote repository**:
6. You will not be allowed since the remote repository is **updated** and the local one is **not**. 
7. After the pull **TortoiseGit** will **try** to pull and merge but it will **fail**, so we have to merge **manually**.
8. Now **resolve the conflict**. Edit the conflicting files and get then correctly merged. Remove all lines that point the locations of the merge conflicts (like <<<<<<< HEAD):
9. Resolve current file with **TortoiseGit** -> **Resolve**



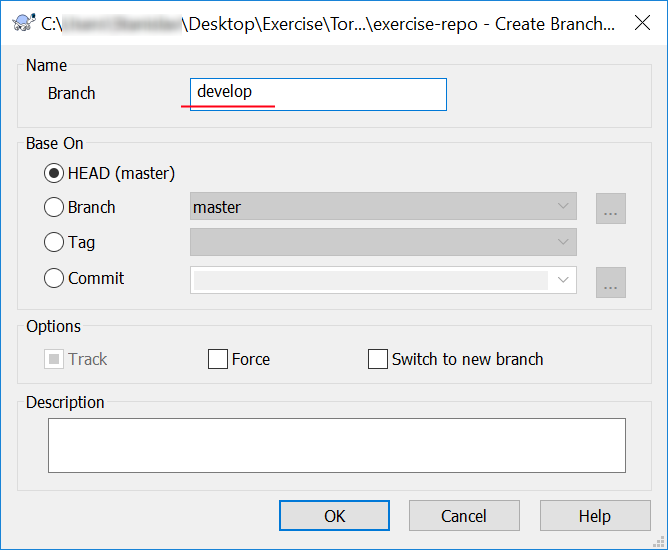
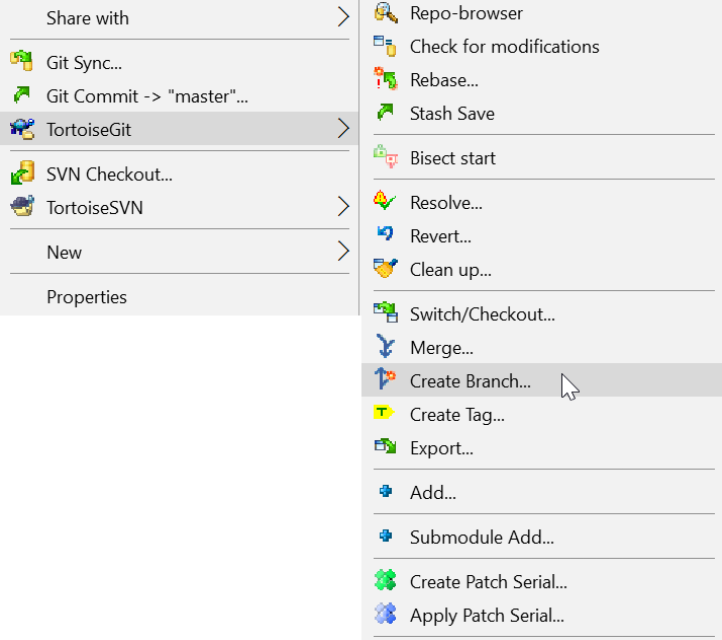
1. Now **commit the merged changes** (your local changes and your changed made from the Web):
2. Now **push again** to push your changes online to GitHub.

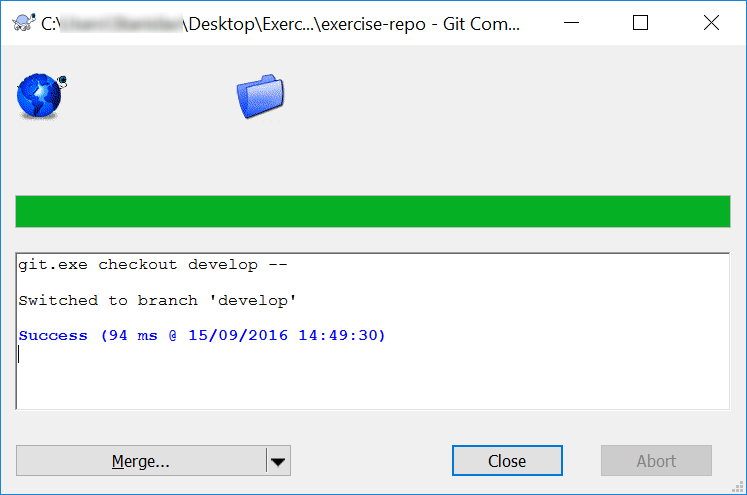
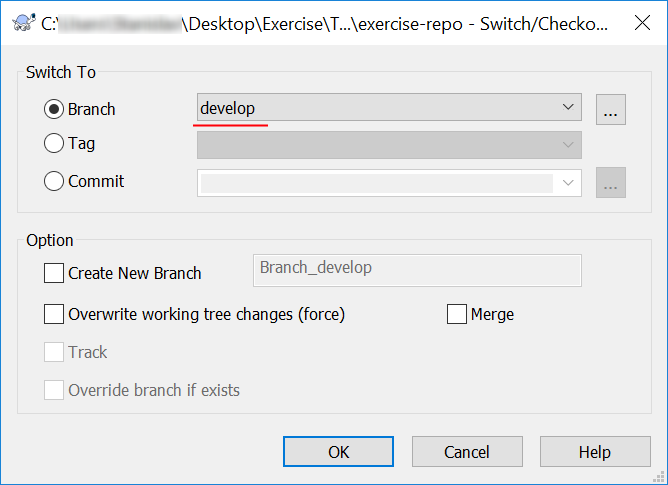
Great, the **push should be successful** with **no conflicts**!

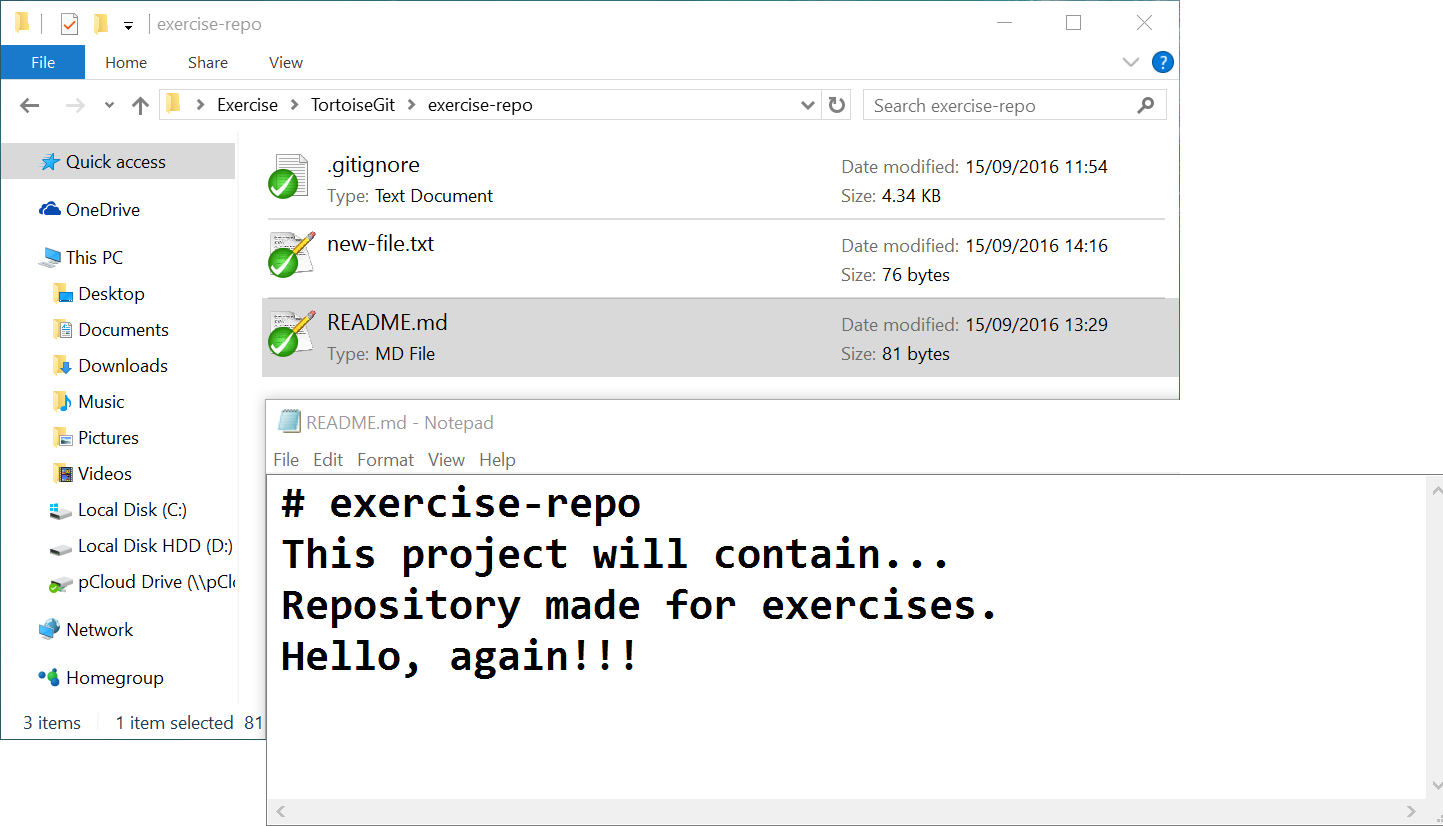
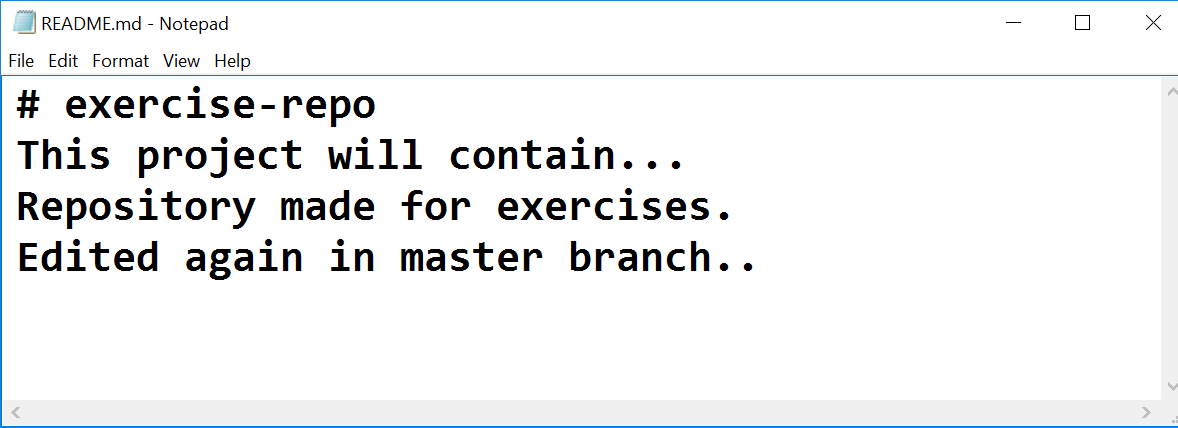
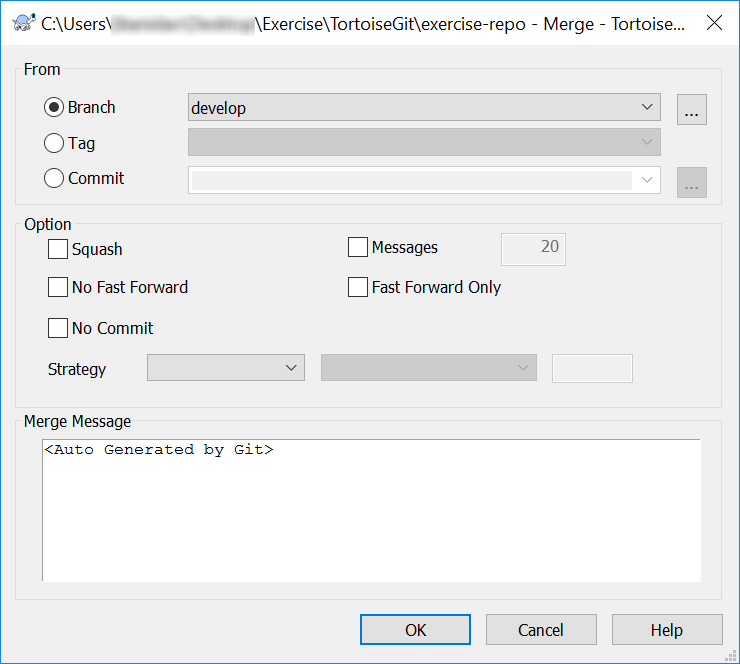
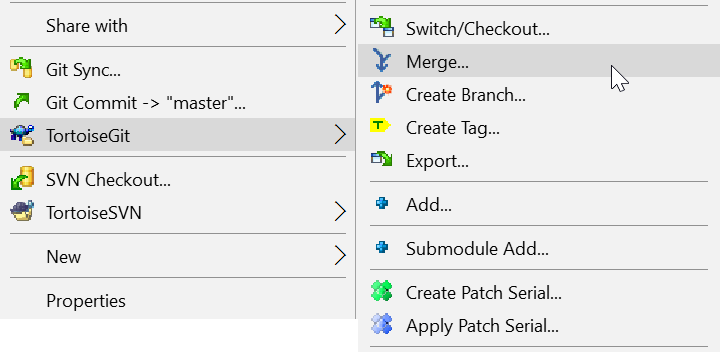
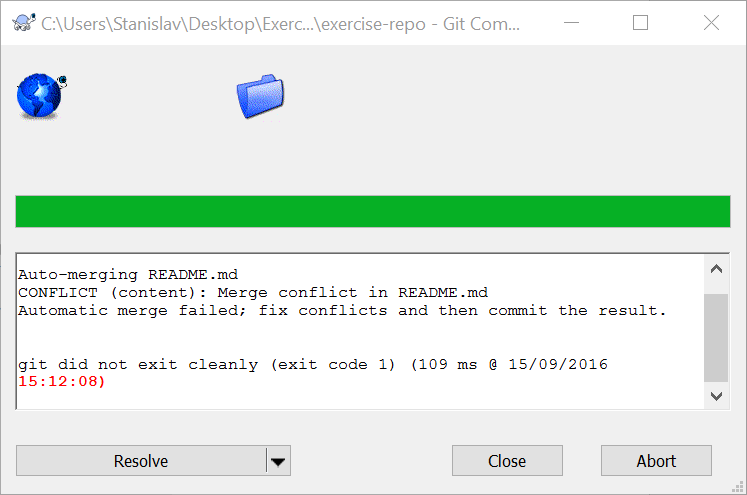
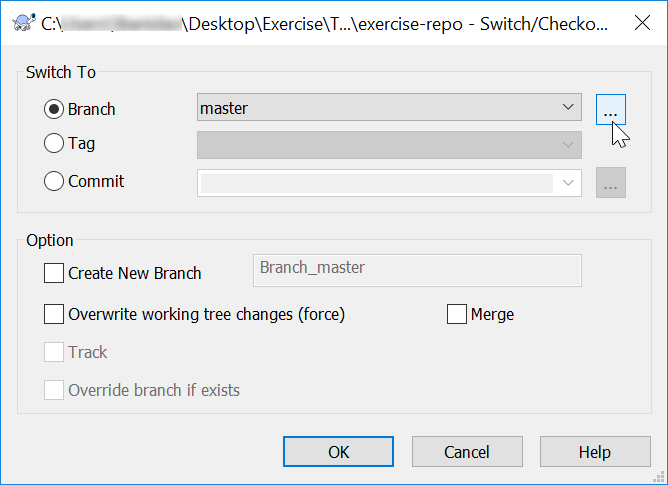
1. Finally, **check the changes** on the Web in your GitHub account:

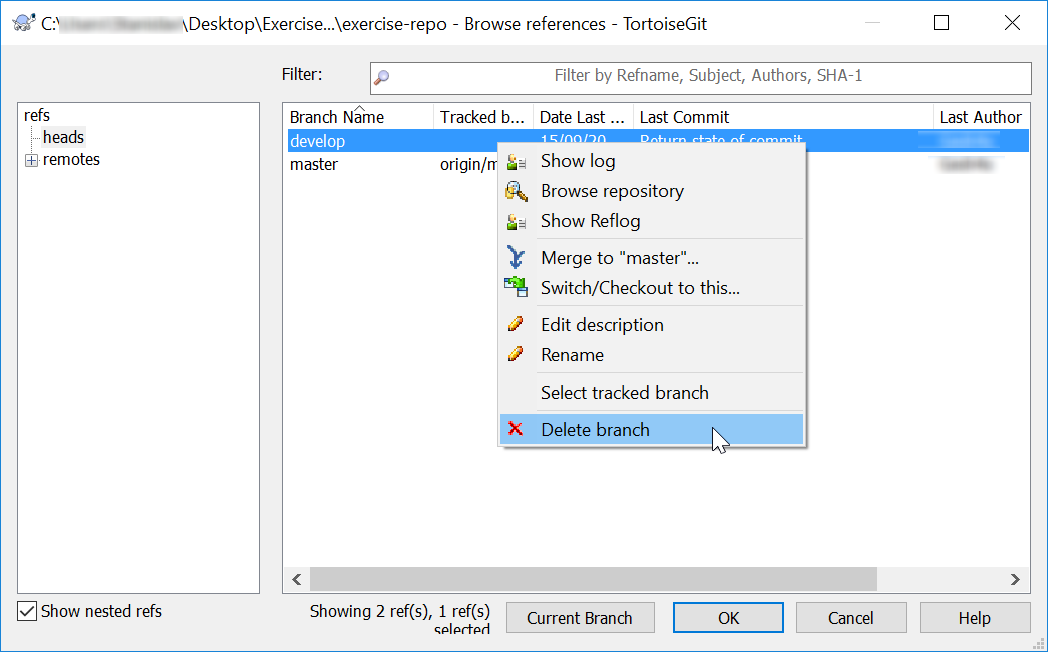


## Create a branch and merge changes to the both local and remote repository

1. ****Create **branch**. (Here the branch name is: **develop**)
2. **Switch** to that branch.

****

1. Make some **changes**. **Edit** one of the files in your repository.
2. **Commit** them.
3. **Switch** to the main branch.
4. Make some changes to the main branch (on the same file you edited before). **Commit** them and then **push**.
5. **Merge** with previous branch (in this case - develop).
6. **Resolve** the new conflicts and commit.
7. **Delete** the newly created branch.
   * Use **TortoiseGit** -> **Switch/Checkout…**

Click on the **hovered** element above and window like this must appear:

You can now **delete** your branch and **commit** your changes.

1. **Update** the remote repository.

# GitBash

## Upload a Few Projects at GitHub

**\*** If you have already cloned your repository with **GitBash** you can safely skip this step.

1. **Clone** the same repository that you worked with for the previous tasks it on your device:

* Use "git clone" command.

1. Open the project files in **Windows Explorer**.
2. Make some **changes** in your favorite text editor:
3. **Commit** your local changes to your local repository.

* Use "git add ." command.
* Use "git commit" command.

1. **Pull** then **push** your changes to the remote repository in GitHub (use):

* Use "git pull" command.
* Use "git push" command.

1. Check whether your changes are online.

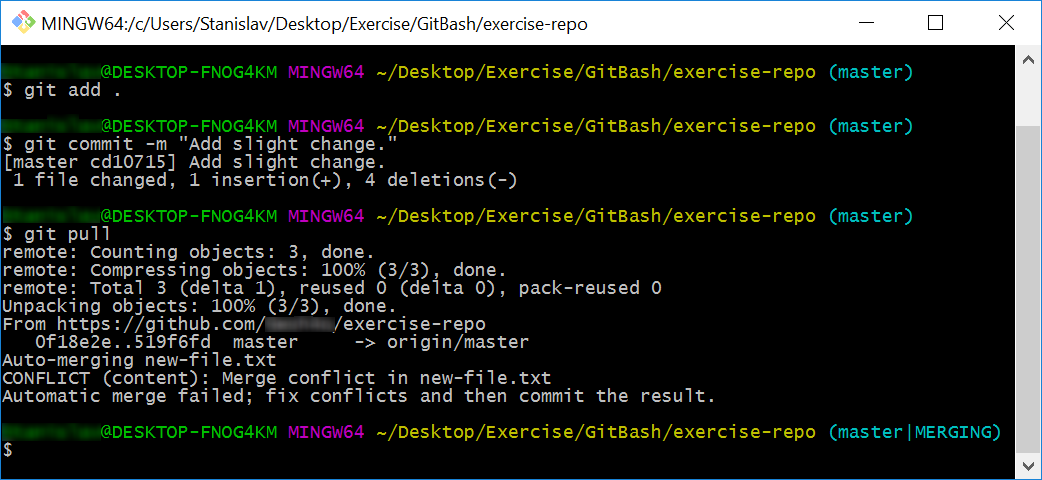
## Make conflicts and resolve them

Create **conflicting changes** and **merge them**. Use the following steps:

1. Make some **changes** in your working directory, e.g. edit the file README.md.
2. **Don’t commit** and **don’t push** your changes yet.
3. Open your GitHub account from your **Web browser** or **TortoiseGit**. Make some changes on the same file.
4. Now **commit** them.
5. Try to **update** the local changes with the **remote repository** at GitHub:

* Use "git pull" command.

1. You will get **conflict notification**.

One of the files from the **local repository** will be **merged** with its newer version from the **remote repository**:

1. Now **resolve the conflict**. Edit the conflicting files and get then correctly merged. Remove all lines that point the locations of the merge conflicts (like <<<<<<< HEAD):
2. Now **commit the merged changes** (your local changes and your changed made from the Web/TortoiseGit):
3. Now **sync again** to push your changes online to GitHub.

Now, the **update should be successful** with **no conflicts**.

1. Finally, **check the changes** on the Web in your GitHub account or sync your TortoiseGit local repo:

## Create a branch and merge changes to the both local and remote repository

1. Create **branch**.

* Use "git branch branchName" command.

1. **Switch** to that branch.

* Use "git checkout branchName" command.

**\* Note** that the previous steps can be done also with the following command:

"git checkout -b branchName"

1. Make some **changes**.
2. **Commit** your changes.
3. **Switch** to the main branch.

\*Look at step №2.

1. Make some changes to the main branch.
2. **Merge** with previous branch.
3. **Resolve** the new conflicts (if any).
4. **Delete** the newly created branch.

* Use "git branch -d branchName" command.

1. **Update** the remote repository.