Git rules

Every repository must have the main branch called **master**

- **master**
    - All tasks that were accepted by reviewers in pull request must be merged into master branch

Every task **must have it's own branch**.
This branch will be identified by speficit format:

- Format = [USER-STORY-XXXX/description]
    - XXXX - unique value generated by TFS
    - description - text from task title separated by "-"

Examples:
USER-STORY-10517/Rewrite-GIT-methodics-to-be-more-understable
USER-STORY-10558/Virtual-reality-module

Workflow

I don't have installed Git Bash

1. Download the **Git Bash** program from page
2. Install **Git Bash**

I have installed Git Bash (Clone repository - first start)

1. Open **Git Bash** program (in Windows called **MINGW64**)
2. Type the command `git config --global http.sslVerify false` to disable the SSL CERT verification, because you can have trouble with pushing to the server
3. Go to required folder, where you want to have your **local git** with command **cd** (Change directory)

Examples:
Change to the required directory with absolute path:
```
C:\School> cd C:\School\3DSpaceGen
```
Change to the parent directory:
```
C:\School> cd ..
```
Change to the grant-parent directory:

`C:\School\backup\January> cd ..\..`

Change to the ROOT directory:

`C:\School\backup\January> cd \`

Display the current directory in the specified drive:

`C:\> cd D:`

3. Go to the TFS Git page and choose your repository (folder/file...) that you want to clone in left side on webpage
4. After choose the repository click to the **Clone** button
5. Copy the displayed url, type `git clone yourcopiedurl` into the **Git Bash** and press **Enter** button
6. **Example:** `git clone http://yourwebpage.com`
7. Now you have actual version of git repository in your PC

Creating branches

1. When you have cloned needed repository, then you need to REBASE your local repository with command `git rebase origin/master` (more about rebase)
2. Now you can create new branch for task with command `git checkout -b [branch_name]`

   Example: `git checkout -b USER-STORY-10558/Virtual-reality-module`
3. Now push your local created branch to TFS Server with command: `git push origin [branch_name]`

   Example: `git push origin USER-STORY-10558/Virtual-reality-module`
4. Type `git rebase origin/master`
5. Type `git push --force`

   Warning: Error message might be occured:

   `fatal: The current branch USER-STORY-10558/Virtual-reality-module has no upstream branch`

   then type: `git push --set-upstream origin USER-STORY-10558/Virtual-reality-module` and repeat the fifth point
6. Now you can add new folder(s)/file(s) with `git add` command

Examples:

Stages all changes:

`git add -A`

Stages new files and modifications, without deletions

`git add .`

Stages modifications and deletions, without new files

`git add -u`

Adding the concrete file/folder:
`git add myfile.txt`
With `git status` command you can verify the current state of your git branch

9. When you want to commit your changes type `git commit -m "#TASK_ID define_your_changes"`
10. **Example**: git commit -m "#10517 Rewriting git methods"
11. **General rule, commit often!!!**

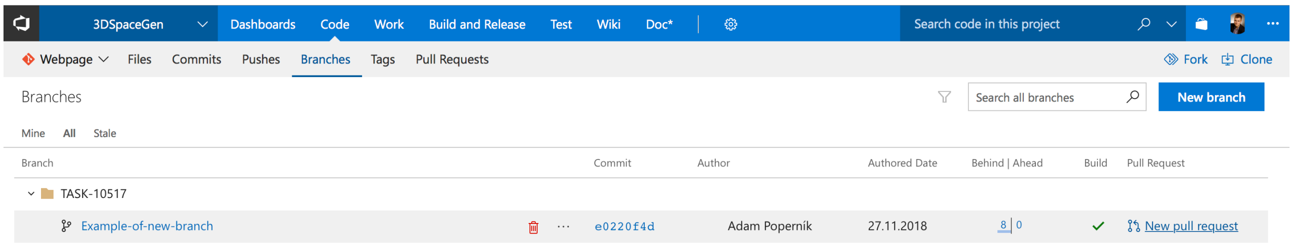
You can use the command `gitk` for displays changes in a repository or a selected set of commits.
Gitk includes visualizing the commit graph, showing information related to each commit, and the files in the trees of each revision.

10. Type git status and check if you have committed all changes
11. Use `git push` to upload the changes into TFS

WARNING - It is necessary to frequently rebase

Pull request

1. Go to the TFS -> Code -> from left side choose your repository
2. Click on the branches button
3. Choose branch which you want for pull request and click on the button **New pull request**



Branch	Commit	Author	Authored Date	Behind | Ahead	Build	Pull Request
TASK-10517						
Example-of-new-branch	e0220f4d	Adam Popernik	27.11.2018	8 | 0	✓	New pull request

4. Set the place where you want to merge your branch - standard set into **master** branch and set the title of the pull request
5. Write the description and add reviewers that you want, then click on the button Create
6. You have created the pull request

All pull requests must be reviewed by at least one other team member. The reviewer might reject the pull request but **must specify the reason**

Conflicts

Sometimes you can have conflicts in your code with master branch (when you don't often rebasing) and when you have changes in your local branch and after a long time you write the command `git rebase`
So you need to resolve the merge conflicts and continue rebasing

1. Merge conflicts you can solve with `git mergetool`, but you need to configure it before first start
2. We will use the kdiff3 as preferred mergetool - click for download

For Windows Users:
```
git config --global --add merge.tool kdiff3
git config --global --add mergetool.kdiff3.path "C:/Program Files/
KDiff3/kdiff3.exe"
git config --global --add mergetool.kdiff3.trustExitCode false

git config --global --add diff.guitool kdiff3
git config --global --add difftool.kdiff3.path "C:/Program Files/
KDiff3/kdiff3.exe"
git config --global --add difftool.kdiff3.trustExitCode false
```

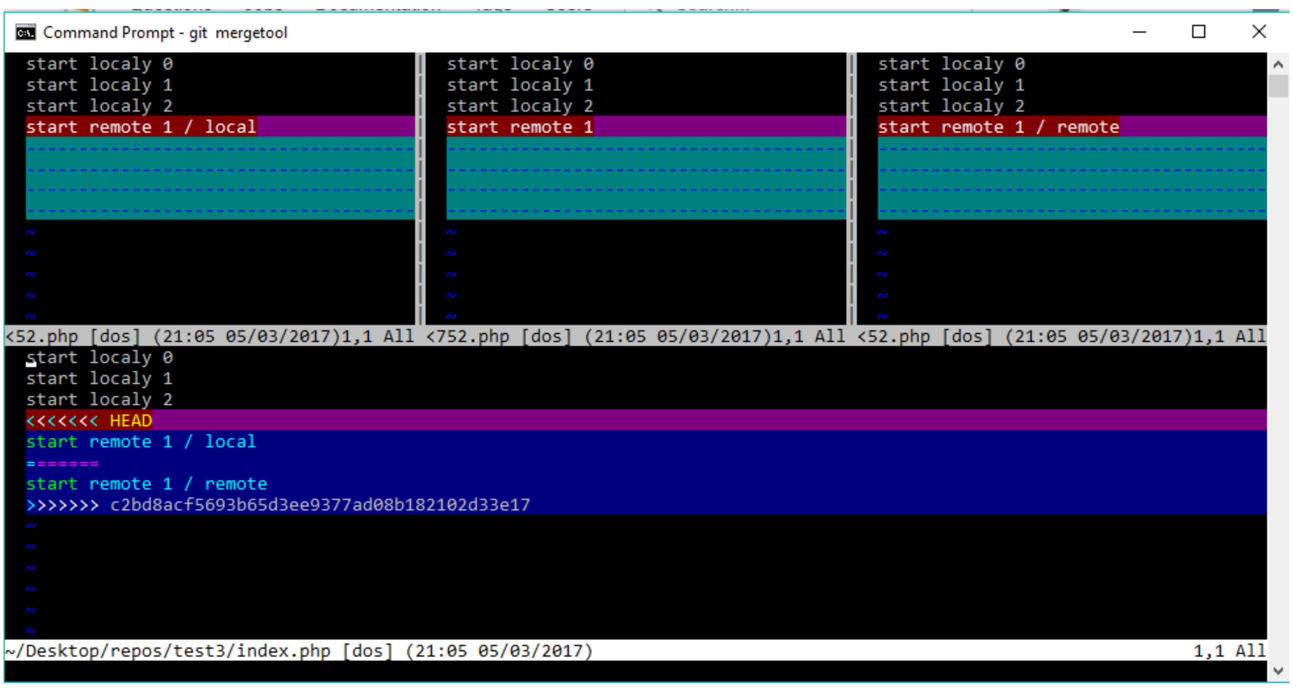
Fow Mac Users:
```
git config --global --add merge.tool kdiff3
git config --global --add mergetool.kdiff3.path  "/Applications/
kdiff3.app/Contents/MacOS/kdiff3"
git config --global --add mergetool.kdiff3.trustExitCode false

git config --global --add diff.guitool kdiff3
git config --global --add difftool.kdiff3.path "/Applications/
kdiff3.app/Contents/MacOS/kdiff3"
git config --global --add difftool.kdiff3.trustExitCode false
```

For Linux Users
```
git config --global --add merge.tool kdiff3
git config --global --add mergetool.kdiff3.path "/usr/bin/kdiff3"
git config --global --add mergetool.kdiff3.trustExitCode false

git config --global --add diff.guitool kdiff3
git config --global --add difftool.kdiff3.path "/usr/bin/kdiff3"
git config --global --add difftool.kdiff3.trustExitCode false
```

2. So when you have installed the program, type the `git mergetool` and merge your conflicts - with choosing the code which will be used and included into merge


Command Prompt - git mergetool

```
start localy 0                    start localy 0                    start localy 0
start localy 1                    start localy 1                    start localy 1
start localy 2                    start localy 2                    start localy 2
start remote 1 / local            start remote 1                    start remote 1 / remote
<52.php [dos] (21:05 05/03/2017)1,1 All <752.php [dos] (21:05 05/03/2017)1,1 All <52.php [dos] (21:05 05/03/2017)1,1 All
start localy 0
start localy 1
start localy 2
<<<<<<< HEAD
start remote 1 / local
=======
start remote 1 / remote
>>>>>>> c2bd8acf5693b65d3ee9377ad08b182102d33e17
~/Desktop/repos/test3/index.php [dos] (21:05 05/03/2017)                                1,1 All
```

3. After merging, you can type `git rebase --continue`
4. Now you have merged branches