# PROJECT CORE PHASE – Traffic Lights System

# Group “E”

Git(Hub) – Good Practices Document

19.02.2016

Created by: Bilger Yahov

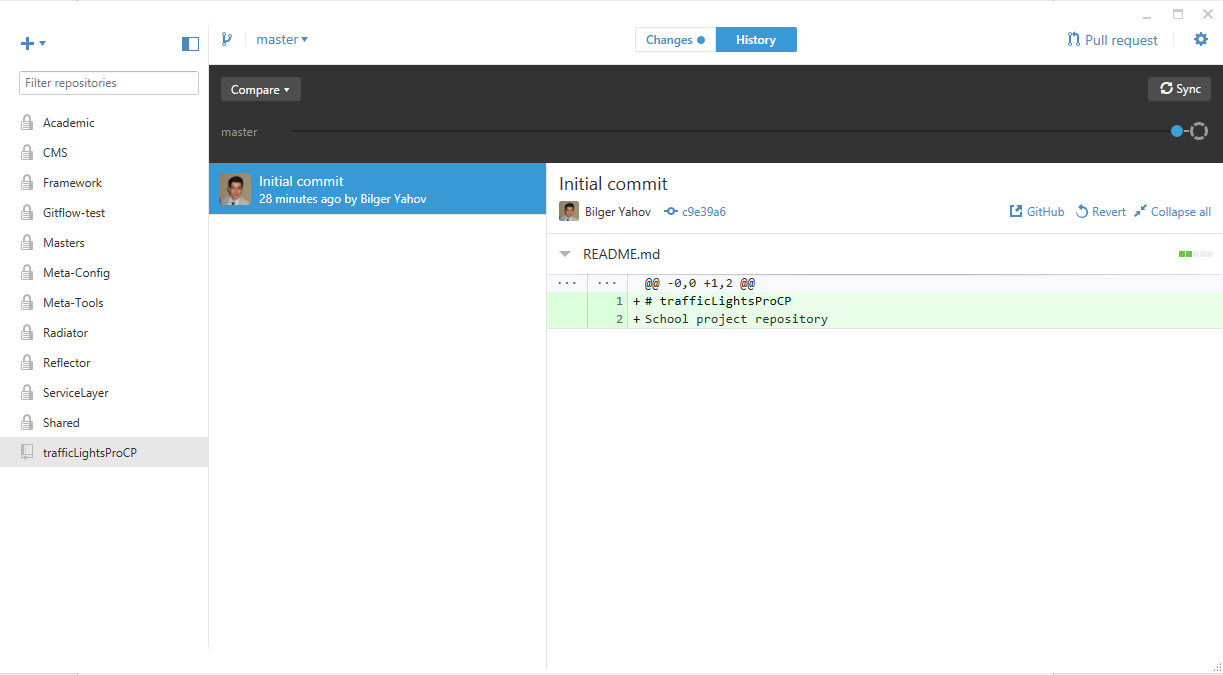
Modified 29.02.2016

1. Download Git for your Operating System from <https://github.com/>
2. Install Git. (If you encounter any difficulties, contact me.)
3. If you don’t have an account – then create one.
4. Clone this repository to a local drive

<https://github.com/bilgeryahov/trafficLightsProCP>

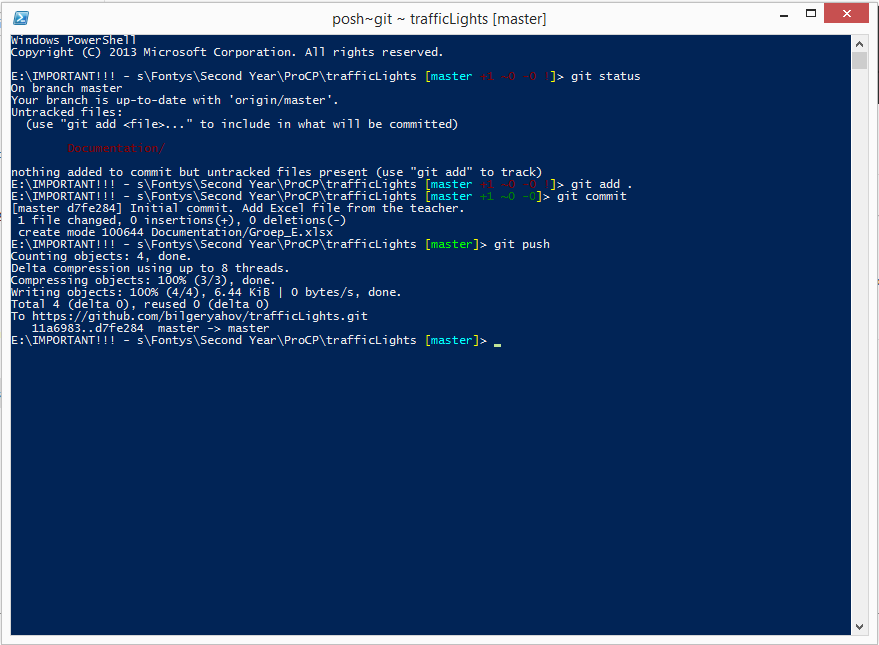
Continues below…

1. After opening Git client you should be able to see this



1. On the left you can see your repositories. Of course not the same as mine. We should have one in common which is “trafficLightsProCP”.
2. In the “History” you can see the history of commits by all the project members or so-called repository contributors.
3. Right click on the “trafficLightsProCP” repository and open the Git Shell.

Continues below….



1. After opening Git Shell you should be able to see this on your screen. We will be using it to run simple operations like commit, push and pull.

Commands:

To pull the last changes from the “Master” branch use command

**git pull --ff-only**  (watch out that before ff there are two dashes!)

The command is self-explanatory and there is no need to explain what it does.

To push your changes to the repository use the commands in this sequence

**git add .** (watch out that after add there is white space and after it comes the dot)

This command adds all the tracked files from your local copy to the “waiting list”.

**git commit**

After this command notepad opens. Without moving the cursor just type your commit message and click “Save” to the file. You do not need to save the file somewhere locally! Just pres Ctrl + C and the magic is done!

This command commits all the changes from the “waiting list”.

**git push**

This is the last command of pushing your local changes to the repository.

This command pushes all the added and committed files to the “Master” branch of the repository.

Continues below….

Here I will give an overview of how to use the commands in a sequence.

After making your changes keep the following steps:

**git stash** (this command parks your changes somewhere aside)

**git pull --ff-only** (this command pulls all the changes from the Repository

**git stash apply** (this command gets your changes back from the parking spot ☺)

**git add .** (WATCH OUT THERE IS WHITE SPACE AND THEN A DOT this command adds all the tracked files to the waiting list)

**git commit** (this command commits all the files)

**git push** (this command pushes all the changes to the master branch)

How to use Git branches:

First of all I would like to emphasize that the workflow will be done in the branch named “Develop”.

In order to go to that branch you only need to type:

**git checkout develop**

What you need to do now immediately and also regularly is to pull everything. Mentioned above how to do that.

As already discussed we will be working with branches about different functional parts of the application. In order to create a branch for yourself type in the Shell:

**git checkout -b feature-bilger develop**

The newly created branch will be starting from develop.

After that you need to push your branch to the repository.

You can do that by typing following in the Shell:

**git push --set-upstream origin feature-bilger**

watch out before set there are two dashes !!!

After that you can start working on your stuff. When you are done just type the commands described above. Git add, git commit git push… Look above!

Always name your branches in the following way:

feature-<the name of the feature>-<your name>

feature-blabla-bilger

When you are done with the particular feature and you want to merge it into develop the things that you need to do:

Checkout to develop,

Pull everything from develop

Go to your branch

Merge develop INTO YOUR BRANCH:

**git merge --no-ff develop**

watch out there are two dashesh before no!!!

After the merging process you have everything from develop merged into your branch. If there are any problems to be fixed please do in your branch.

After fixing the problems do the steps: git add, git commit, git push.

This is to update your branch which now is the newest one. With everything from develop merged with your branch and fixed errors.

When you are done with this go to develop again, pull everything if there are new things that came in do the same steps as above or if everything is up-to-date do:

**git merge --no-ff <your branch name>**

After merging you will see that the develop sign font gets green. That means that you need to push.