Gerrit
Quick start
v1.0.2

Paweł Warzecha

November 1, 2018

Contents

1 Introduction

Gerrit is a free, web-based team code collaboration tool. Software developers in a team can review each other's modifications on their source code using a Web browser and approve or reject those changes. It integrates closely with Git, a distributed version control system. Gerrit is a fork of Rietveld, another code review tool. Both namesakes are of Dutch designer Gerrit Rietveld.[1]

2 Using Gerrit

For this project the GerritHub[2] is used. It offers free and preconfigured Gerrit server. It has been chosen because it is a ready to use solution, widely used and integrated with the GitHub.

Next sections show how to use Gerrit.

2.1 Registering to GerritHub.io

To use the GerritHub, the GitHub account is needed. If you don't have a GitHub account, create it first. Next if You don't have a GerritHub account click the "First time Sign In" button (marked with the red rectangle in the figure 1). Next follow the instructions on the website.

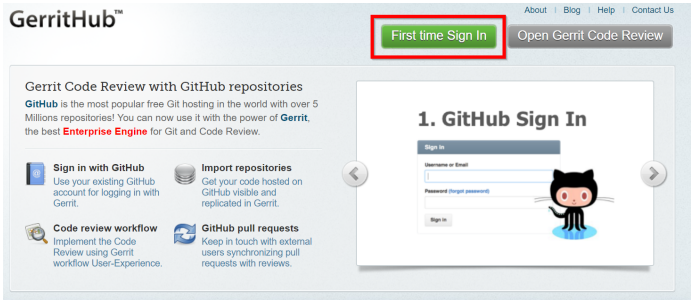

Figure 1: Connecting github.

2.2 Cloning repository

To clone the repository with the project code go to Browse->Projects and in the Filter write the name of the project in which you want to contribute (as shown in figure 2). Next click on the project name.

On the site you can see the command which is needed to clone the repository (marked in figure 3)

Figure 2: Finding a project.

Figure 3: Clone command.

2.3 Pushing changes

If you want to push your changes to review, just use the git command, but change the branch name in the command to *HEAD:refs/for/<branch_name>* . For example, if you want to push the changes to a master branch use:

```
git push origin HEAD:refs/for/master
```

2.3.1 Adding reviewer

There are two ways to add reviewer(s) to your change. First, assign the reviewer at the push. In this case use *%r=<email_address>* at the end of the push command. The command may look like:

```
git push origin HEAD:refs/for/master%r=ProrokWielki@o2.pl
```

The second way is to set the reviewer on the web site. In this case from *YOUR->Changes* select the change to which you want to assign the reviewer. Then on the web page click *ADD REVIEWER*. Put the reviewer email next to the *Reviewers* (as shown in figure 4).

2.3.2 Adding status

The change can be marked as work in progress (*WIP*). It indicates the change is not yet finished. To set the state to *WIP* you can add the *%wip* to push command:

```
git push origin HEAD:refs/for/branch_name%wip
```

You can also set it from the web page. Go to the change and click *WIP* in the *MORE* menu, as shown in figure 5.

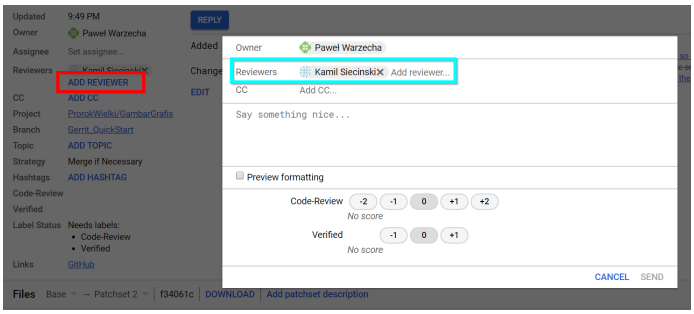

Figure 4: Adding Reviewer.

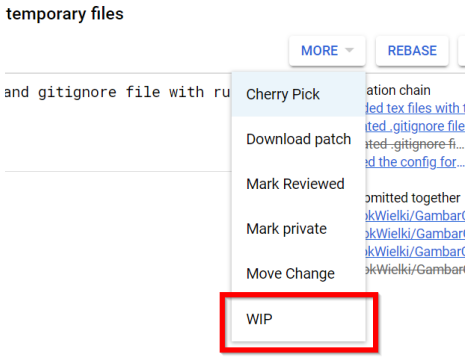

Figure 5: Setting WIP state.

2.4 Reviewing changes

If you are asked to review someones change you can see that change in the *Incoming reviews* in the *YOUR->Changes* section.

In the Files Section you can find the files which were pushed with the change. To see the content of a file, click on its name.

If you want to add comment to a part of the file, select the text to comment and press *c*. After writing the comment click submit.

After reviewing all files you can mark the change. Click on the *REPLY* button and score the change with points(as shown in figure 6).

To finish click *SEND*

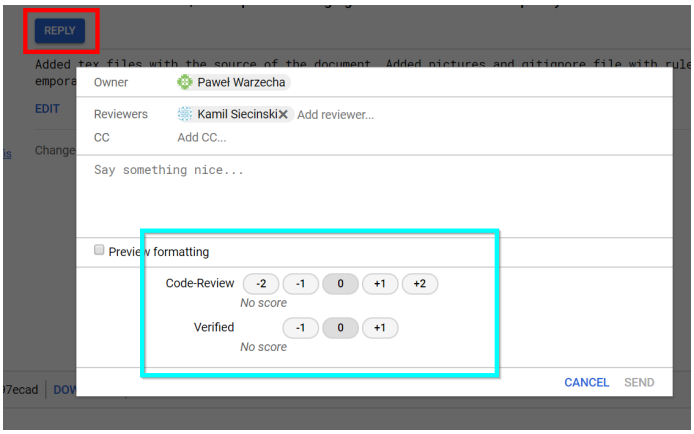

Figure 6: Marking a change.

2.5 Updating change

After your change was reviewed, it might happen you need to update something in the change.

First of all you have to download your change. To do that, on the web page, go to the change. From the *MORE* menu select *Download patch*. From there copy *Checkout* command and run it (shown in figure 7).

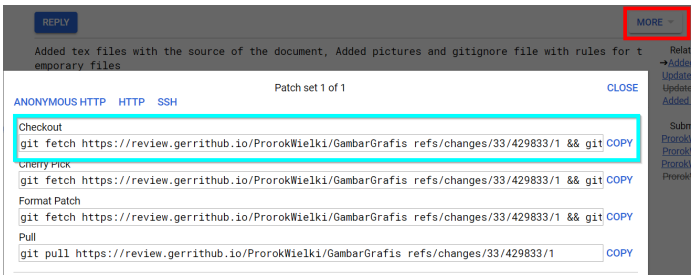

Figure 7: Checking out a change.

Next make necessary changes to the files. After that when committing the change use *–amend* option. And in the comment add the *Change-Id*(as shon in figure 8) associated with the change(can be found on the change web page figure 9).

```
git commit ––amend
```

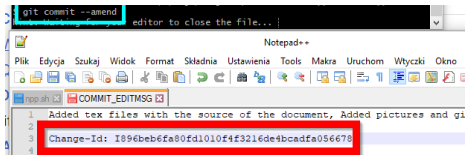

Figure 8: Adding change id.

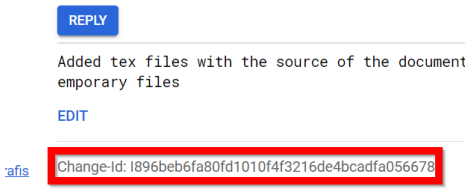

Figure 9: Change-Id localization.

After that push the change.

2.5.1 Changing status to ready

If your change was in *WIP* status, and after updating the change it is ready to be reviewed add *%ready* to the push commend.

```
git push origin HEAD:refs/for/branch_name%ready
```

References

[1] *Wikipedia – Gerrit (software).*
`https://en.wikipedia.org/wiki/Gerrit_(software)`

[2] *GerritHub.io.*
`http://gerrithub.io/`

Revision History

Revision	Date	Author(s)	Description
1.0.0	18.10.2018	PW	Created
1.0.1	31.10.2018	PW	After review – typos fixed
1.0.2	1.11.2018	PW	After review – one more typo fixed