git advanced - Labs

Learning-Catalogue code: 070174

Sébastien SNAIDERO

Software product owner
& senior developer
@ TDS / PDK & Design Flows

life.augmented

Lab : workflow

Lab – workflow (1)

- Start
  - Open git bash
  - Create a new folder named “git_labs” and go into

- Clone the workflow repository

  ssh://gitolite@codex.cro.st.com/git-trainings/Advanced/Workflow.git

- Let’s say your task name is “art #395473: implement feature 1”
  1. Checkout a new task branch name with the task id and a short descriptive title
     - git checkout -b feature/395473-implement-feature1
       - The ID to easily associate the track with its tracker
       - The description if for a human little hint on what’s in it
  2. Do you work on this branch
     - Into users/<yourName> : Perform **four** Commits of your choose (change 1, change 2..)
  3. use interactive rebase to squash all commits together
     - git rebase -i master

Lab – workflow (2)

- git will display an editor window with lists of your commits
    - pick 3dcd585 Adding Comment model, migrations, spec
    - pick 9f5c362 Adding Comment controller, helper, spec
    - pick 977a754 Comment belongs to a User
    - pick 9ea48e3 Comment form on Post show page
  - Now we tell git what we want to do (squash)
    - pick 3dcd585 Adding Comment model, migrations, spec
    - squash 9f5c362 Adding Comment controller, helper, spec
    - squash 977a754 Comment belongs to a User
    - squash 9ea48e3 Comment form on Post show page
  - Save and close the file
    - This will squash all commit together into one commit
- Git displays a new editor where we can **give the new commit a clear message**
  - Message must be written on the first line (lines after are commit message details)
  - We will use the task ID and tile : **art #395473: implement feature 1**
  - Save and close the editor

life.augmented

Lab – workflow (3)

7. Merge your changes back into master
   - git checkout master
   - git merge feature/395473-implement-feature1
     - It must be a fast-forward merge
8. Finally push your change to upstream
   - If, meanwhile origin is updated do:
     - git fetch origin
     - git rebase origin/master
9. Use gitk --all to observe the result

life.augmented

Lab : reset & revert

Reset & Revert

- Revert
  - Clone the following repo

    ssh://gitolite@codex.cro.st.com/git-trainings/Advanced/Revert_Reset.git
  - Use git show to see last commit ID content
  - Use revert command to revert the last commit
  - Use git show to see the reverted commit content

- Reset
  - Use git reset to get back to the first commit
    - Observe the working tree status
  - use reset --hard to get back to initial state (where origin/master is)
    - Observe log & status output
  - Use reset --soft to get back to a previous commit
    - Observe log & status output

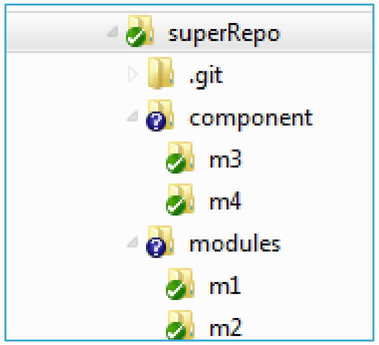
Lab: submodules

Git submodule lab

- Submodules creations
  - Clone the following repo

```
git clone --branch step1 ssh://gitolite@codex.cro.st.com/git-trainings/Advanced/Submodules/main.git step1
```

      - It contains a readme.txt file
  - Use git submodules to add the 4 git repositories contained in the readme.txt
    - The arborescence should be as following →
    - Observe .gitmodules and folders creation.
      - Observe .gitmodules and folders content
    - Use git status to observe the addition
    - git config --global status.submoduleSummary true
    - Use git status to observe changes in output
    - Commit the submodules addition



```
superRepo
  .git
  component
    m3
    m4
  modules
    m1
    m2
```

- Submodules modification
  - Go to modules/m2 & modify file1.txt
    - Observe the independency of the m2 repository
    - Go back to main repo
      - Use status to observe how modification is managed
    - Go to modules/m2 & Commit m2 modifications
    - Go back to main repo & Commit the module modification
    - Use git show to discover what had finally changed in top module

Git submodule lab 2

- Submodules update / clone
  1. Leave the actual git project
  2. Clone the following repo

     `git clone --branch step2 ssh://gitolite@codex.cro.st.com/git-trainings/Advanced/Submodules/main.git step2`
  3. Observe modules' folder content
  4. subModules content must be updated manually:
     - Shortcut #1 : git submodule update --init
     - Shortcut #2 : git clone --recurse-submodules <url>
       - You can Try it: leave the current folder and clone again

- The detached head
  - Move to modules/m1,
    - Observe the « detached head » state
      - Use log command to view the module evolution
      - Use command: git checkout master to update the module
    - Go to main repo: Use status command and observe the working tree situation
    - Commit the new situation

life.augmented

Lab 3: subtree

life.augmented

Subtree lab

- Clone the parent repo

  ssh://gitolite@codex.cro.st.com/git-trainings/Advanced/Subtrees/parent.git

  - Use log command to observe its commit
  - Create a branch named with your shortlogin
    - git checkout -b <login>
  - Push the branch to remote
    - Git push origin <login>

- Leave the folder & Clone child repo and look to its commits

  ssh://gitolite@codex.cro.st.com/git-trainings/Advanced/Subtrees/child.git

  - Use log command to observe its commit
  - Create a branch named with your shortlogin
    - git checkout -b<login>
  - Do 2 commits on the created branch
  - Push the branch to remote
    - Git push origin <login>

- Back to parent repo (Ensure to work on your branch)
  - Use subtree to add the child's created branch (prefix it "my-child")

    ```
    git subtree add --prefix=my-child ssh://gitolite@codex.cro.st.com/git-trainings/Advanced/Subtrees/child.git <login>
    ```

    - use gitk to observe the results

life.augmented

Subtree lab

- Changing the child project from parent
  - Add a file to “my-child” folder & commit the modification & push your branch
    - Use gitk to observe the results
  - Go to the child folder
    - Observe that it has no changes
  - Back to parent folder

- Contribute the change to child repository
  - On parent folder use git subtree push

    ```
    git subtree push --prefix=my-child ssh://gitolite@codex.cro.st.com/git-trainings/Advanced/Subtrees/child.git <login>
    ```

    - Use gitk to observe the results
  - Back to child folder
    - Pull your branch and observe the results

- Bring updates from child to parent
  - Perform a commit on child and push your branch
  - Back to parent and use git subtree pull
    - Use gitk to observe the results
      - Observe how your commit is duplicated

life.augmented

Subtree lab

- Additional : Bring updates from child to parent + squash
  - Perform **3** commits on child and push your branch
  - Back to parent and use git subtree pull with squash option

```
git subtree pull --prefix=my-child ssh://gitolite@codex.cro.st.com/git-trainings/Advanced/Subtrees/child.git <login> --squash
```

    - Use gitk to observe the results

life.augmented

Lab : Repo

Repo lab – Install repo & python

- Install Repo

```
#Create path if not present & clone repo-tool
mkdir -p /c/git
cd /c/git
git clone ssh://gitolite@codex.cro.st.com/git-trainings/git-repo-official.git /c/git/git-repo
cd git-repo
git checkout stable
```

- Install python
  - Install python 2.7.13 (or Anaconda) and the pywin 2.7 library

- Add binaries to your paths

```
#Create the file .bashrc in your HOME directory if not present
cd ~
pwd
touch .bashrc
```

  - Add the following lines to the created bashrc file

```
PATH="/c/Python27":$PATH
PATH="/c/git/git-repo":$PATH
```

Repo lab – Install repo under Unix

- Install Repo

```
#Create path if not present & clone repo-tool
mkdir -p <root-path>
cd <root-path>
git clone ssh://gitolite@codex.cro.st.com/git-trainings/git-repo-official.git
cd git-repo
git checkout stable
```

- Load python

```
sw python 2.7.15
```

- Add binaries to your paths

```
setenv PATH "<root-path>/git-repo:${PATH}"
```

Repo lab

- Observe our manifest file
  - clone the following repo
    - git clone ssh://gitolite@codex.cro.st.com/git-trainings/Advanced/Repo/manifest.git
  - Observe the default.xml manifest file

- Download our project through repo command
  - Create a new folder
  - Initialize repo
    - repo init
      - Observe the .repo folder creation
  - Initialize repo + Specify the manifest url
    - repo init **-u** ssh://gitolite@codex.cro.st.com/git-trainings/Advanced/Repo/manifest.git
  - Use repo sync command to download the project
    - This will **clone** all projects named into manifest to the path specified
    - Explore the 4 repositories cloned and observe the “detached HEAD” state

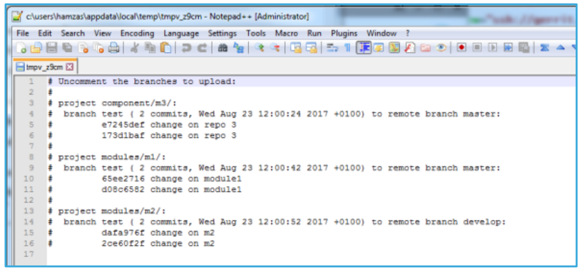
Repo lab

- Start working, create a branch
  - To create a branch on different repository
    - repo start **lab** modules/m1 modules/m2
      - Repo start will create a branch named lab in repository on modules/m1 & modules/m2
    - repo start --all <login>
  - Use repo branch to confirm branches creation
    - Can be run from any directory under the root directory
  - Use repo checkout <login> to switch all your repositories to <login>

- From here we can work normally using git
  - We will, on 2 repositories modify files, view status, view diff, commit
    - Modify component/m3/file & component/m4/file4.txt
    - Use repo status to list the states of your files
      - Modify files on both repositories
    - Use repo diff to see uncommitted edits
    - Commit the diff using git command (each repository separately)
  - The push can be done from each repo separately (don’t do it)

Repo lab

- Uploading changes
  - Use repo push command to push changes
    - Your default git editor is opened


c:\users\hamzas\appdata\local\temp\tmpv_z9cm - Notepad++ [Administrator]
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
tmpv_z9cm

```
# Uncomment the branches to upload:
#
# project component/m3/:
#  branch test ( 2 commits, Wed Aug 23 12:00:24 2017 +0100) to remote branch master:
#         e7245def change on repo 3
#         173d1baf change on repo 3
#
# project modules/m1/:
#  branch test ( 2 commits, Wed Aug 23 12:00:42 2017 +0100) to remote branch master:
#         65ee2716 change on module1
#         d08c6582 change on module1
#
# project modules/m2/:
#  branch test ( 2 commits, Wed Aug 23 12:00:52 2017 +0100) to remote branch develop:
#         dafa976f change on m2
#         2ce60f2f change on m2

```

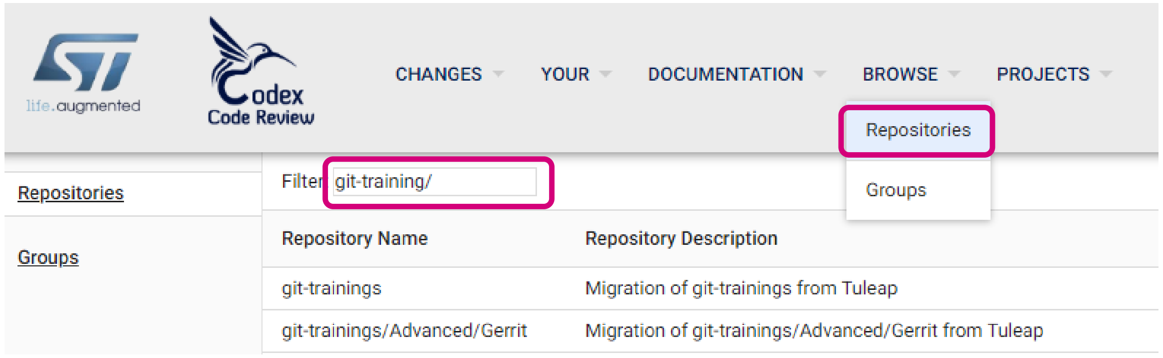
      - Uncomment the branches to upload
      - Save & close
    - This will push the modification
    - repo upload would have send commits for review (**gerrit**), but not set
  - You can push changes on each repository separtely with the traditional git command:
    - git push <remote> <branch>

- Additional command
  - repo list ;

Lab : Gerrit

Gerrit lab

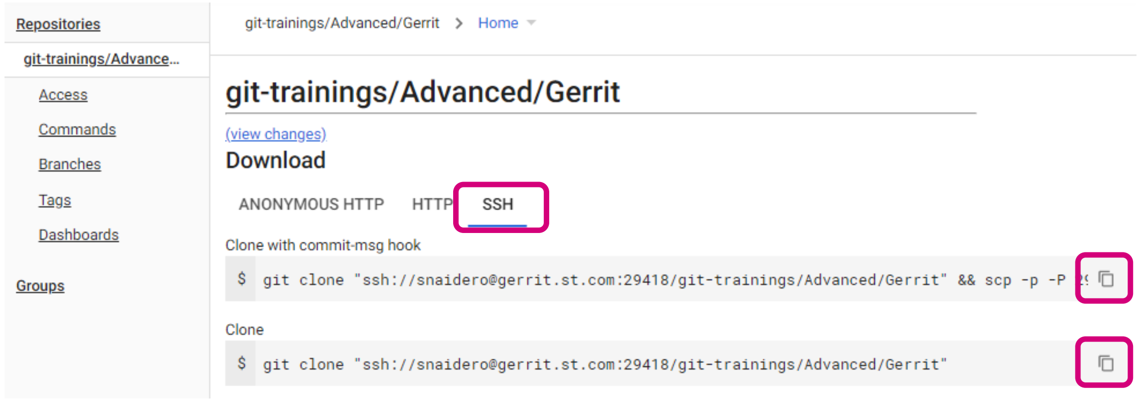
- Clone gerrit repo
  - Open gerrit web interface and sign in
    - https://gerrit.st.com/
    - Click on Browse > Repositories
      - On "filter" field seach for: git-trainings/Advanced/Gerrit



Filter git-training/

Repository Name	Repository Description
git-trainings	Migration of git-trainings from Tuleap
git-trainings/Advanced/Gerrit	Migration of git-trainings/Advanced/Gerrit from Tuleap

  - Click the repo link & copy the clone command



Clone with commit-msg hook

```
$ git clone "ssh://snaidero@gerrit.st.com:29418/git-trainings/Advanced/Gerrit" && scp -p -P 2
```

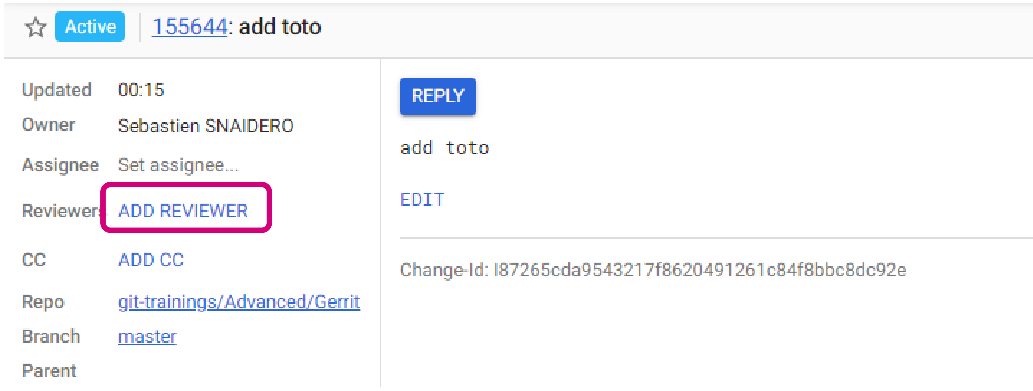
Clone

```
$ git clone "ssh://snaidero@gerrit.st.com:29418/git-trainings/Advanced/Gerrit"
```

Gerrit lab ; as a developer

- Open git bash and paste the command to clone the gerrit repository
  - git clone "ssh://snaidero@gerrit.st.com:29418/git-trainings/Advanced/Gerrit"
  - scp -p -P 29418 snaidero@gerrit.st.com:hooks/commit-msg "Gerrit/.git/hooks/"

- Into users/<yourLogin>
  - Perform a commit
  - push to refs/for/master ;
    - git push origin HEAD:refs/for/master
  - observe log specifc lines
    - remote: **New Changes**:
    - remote: **https://gerrit.st.com/xxxxxx**
      - **Ctrl + click to follow the link (ensure it's opened on chrome)**
  - Add reviewers
    - Mail will be sent to reviewers
    - Observe your received mails.



155644: add toto

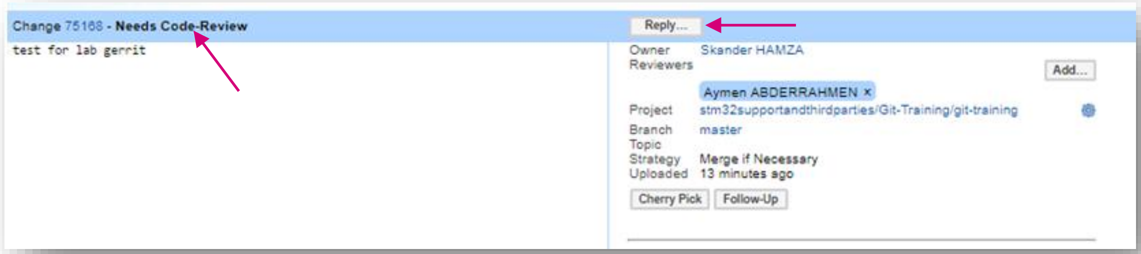
Updated	00:15
Owner	Sebastien SNAIDERO
Assignee	Set assignee...
Reviewers	ADD REVIEWER
CC	ADD CC
Repo	git-trainings/Advanced/Gerrit
Branch	master
Parent	

```
add toto
```

Change-Id: I87265cda9543217f8620491261c84f8bbc8dc92e

Gerrit lab ; as a reviewer

- As a reviewer experiment the following on different patch-sets

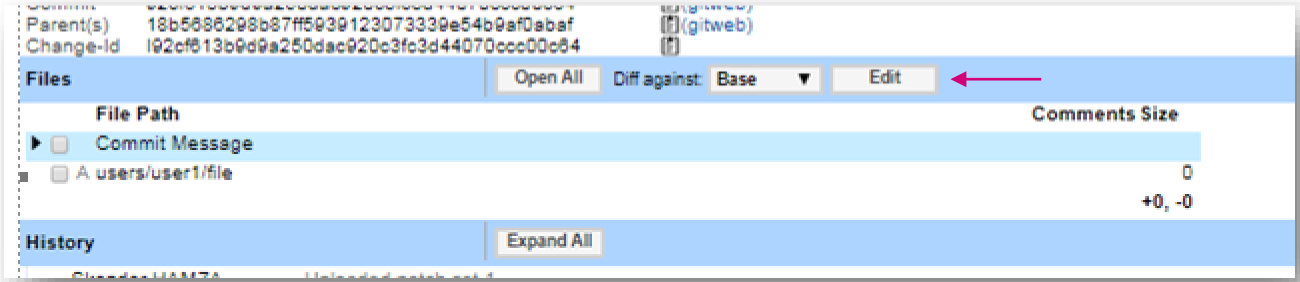


Change 75168 - Needs Code-Review

test for lab gerrit

Owner	Skander HAMZA
Reviewers	Aymen ABDERRAHMEN ×
Project	stm32supportandthirdparties/Git-Training/git-training
Branch	master
Topic	
Strategy	Merge if Necessary
Uploaded	13 minutes ago

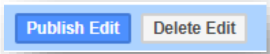
  - Add comment & reply -1
  - Write message & reply +1
  - Edit commit content

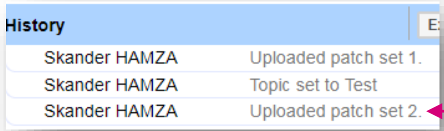


```
Parent(s)   18b5686298b87ff5939123073339e54b9af0abaf   (gitweb)
Change-Id   I92cf613b9d9a250dac920c3fc3d44070ccc00c64
```

File Path	Comments	Size
Commit Message		
A users/user1/file		0
		+0, -0

    - Edit files (add ; rename; edit content) ; edit commit msg ;
        - Save and close
    - Click on done editing
    - Publish edit
    - Observe the new patch set creation.



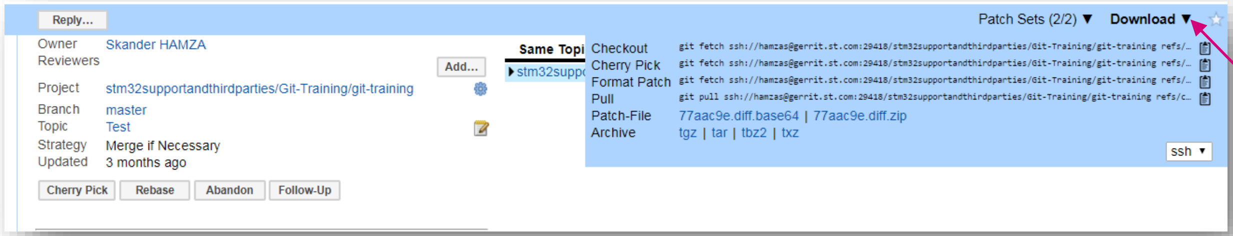


History

Skander HAMZA	Uploaded patch set 1.
Skander HAMZA	Topic set to Test
Skander HAMZA	Uploaded patch set 2.

Gerrit lab ; as a developer

- Consider all your review
  - Get the new patch sets if any

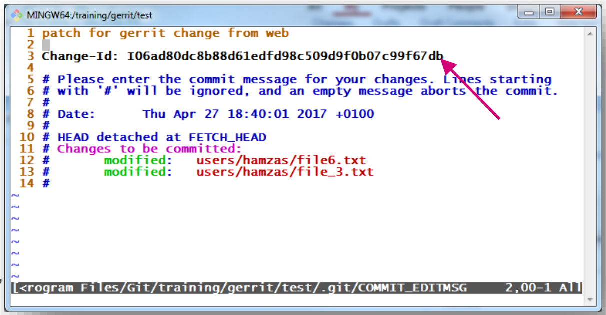


Owner	Skander HAMZA
Reviewers	
Project	stm32supportandthirdparties/Git-Training/git-training
Branch	master
Topic	Test
Strategy	Merge if Necessary
Updated	3 months ago

```
Checkout      git fetch ssh://hamzas@gerrit.st.com:29418/stm32supportandthirdparties/Git-Training/git-training refs/…
Cherry Pick   git fetch ssh://hamzas@gerrit.st.com:29418/stm32supportandthirdparties/Git-Training/git-training refs/…
Format Patch  git fetch ssh://hamzas@gerrit.st.com:29418/stm32supportandthirdparties/Git-Training/git-training refs/…
Pull          git pull ssh://hamzas@gerrit.st.com:29418/stm32supportandthirdparties/Git-Training/git-training refs/c…
Patch-File    77aac9e.diff.base64 | 77aac9e.diff.zip
Archive       tgz | tar | tbz2 | txz
```

  - Use checkout option for instance
    - Observe the detached head; you can create a branch for safety
  - Do all modifications ; apply all comments

- Amend the commit
  - Git commit -a --amend
  - ADD Change ID on message commit details
    - Copy it from web interface
    - Change-Id: I2a3c51dd42e9e1224c0a16fcbb2388e7a597

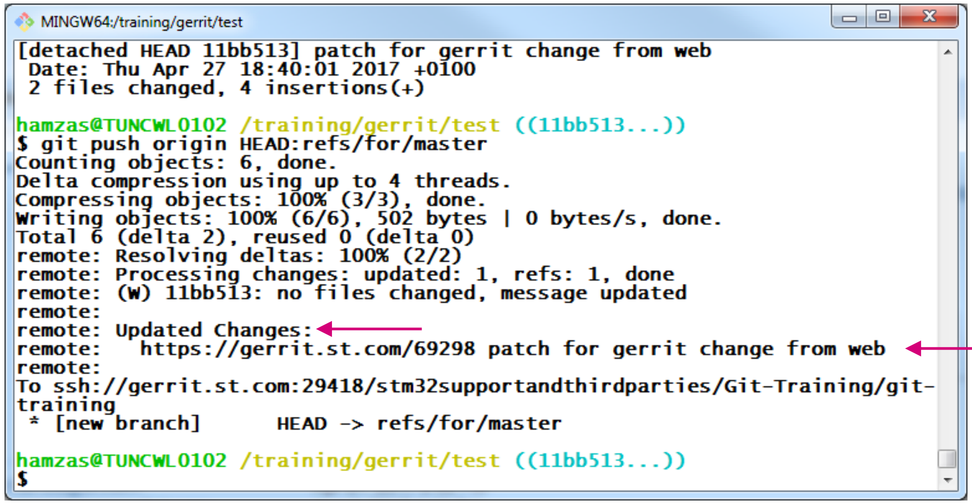


```
patch for gerrit change from web

Change-Id: I06ad80dc8b88d61edfd98c509d9f0b07c99f67db

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Date:      Thu Apr 27 18:40:01 2017 +0100
#
# HEAD detached at FETCH_HEAD
# Changes to be committed:
#       modified:   users/hamzas/file6.txt
#       modified:   users/hamzas/file_3.txt
#
~
~
~
~
~
~
~
~
[<rogram Files/Git/training/gerrit/test/.git/COMMIT_EDITMSG     2,00-1 All
```

- Push the amended commit to refs/for/master
  - Git push origin HEAD:refs/for/master ; the review

Gerrit lab ; as a developer (2)

- Observe the push message to ensure it's an updated patch-set

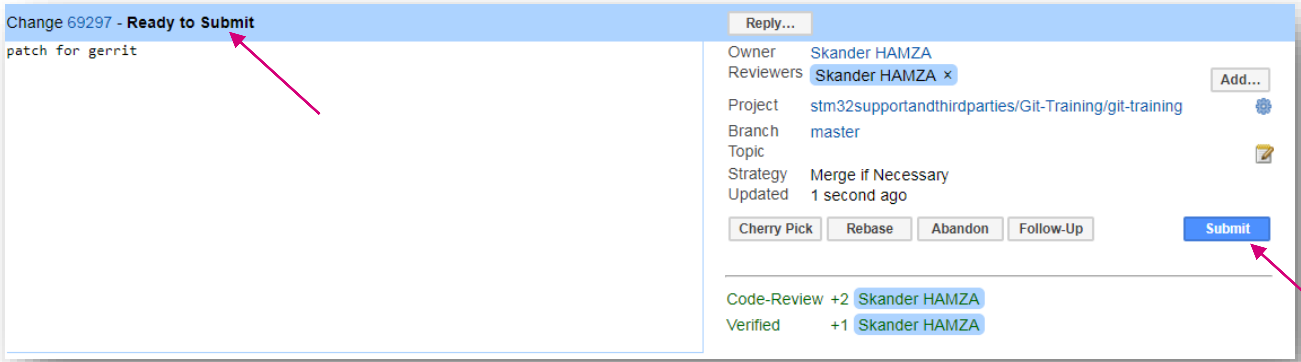


```
[detached HEAD 11bb513] patch for gerrit change from web
 Date: Thu Apr 27 18:40:01 2017 +0100
 2 files changed, 4 insertions(+)

hamzas@TUNCWL0102 /training/gerrit/test ((11bb513...))
$ git push origin HEAD:refs/for/master
Counting objects: 6, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (6/6), 502 bytes | 0 bytes/s, done.
Total 6 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2)
remote: Processing changes: updated: 1, refs: 1, done
remote: (W) 11bb513: no files changed, message updated
remote:
remote: Updated Changes:
remote:   https://gerrit.st.com/69298 patch for gerrit change from web
remote:
To ssh://gerrit.st.com:29418/stm32supportandthirdparties/Git-Training/git-
training
 * [new branch]      HEAD -> refs/for/master

hamzas@TUNCWL0102 /training/gerrit/test ((11bb513...))
$
```

- Observe the messages
  - Updated changes
  - The review url is the same
    - Ctrl+click to follow the link
    - Or refresh the web page

Gerrit lab ; as reviewers / commiter

- Evaluate the reviews and verifications
- Approve the patch-set (+2)
- Oberserve the submit button & the ready to submit state



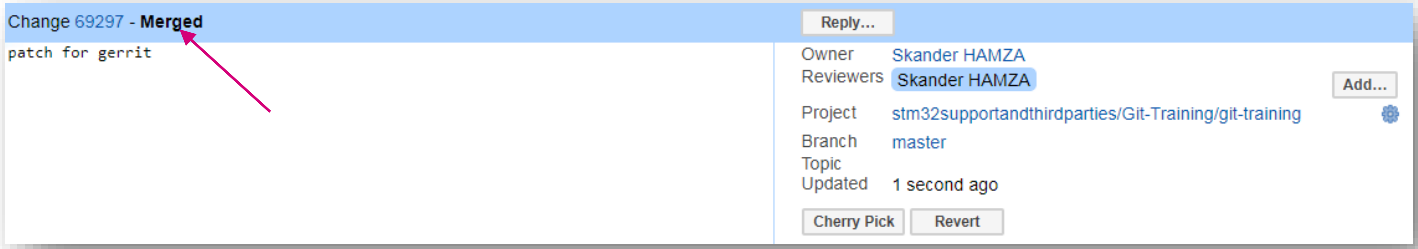
Change 69297 - **Ready to Submit**

```
patch for gerrit
```

Owner	Skander HAMZA
Reviewers	Skander HAMZA ×
Project	stm32supportandthirdparties/Git-Training/git-training
Branch	master
Topic	
Strategy	Merge if Necessary
Updated	1 second ago

Code-Review +2 Skander HAMZA
Verified +1 Skander HAMZA

- Submit; our change is merged



Change 69297 - **Merged**

```
patch for gerrit
```

Owner	Skander HAMZA
Reviewers	Skander HAMZA
Project	stm32supportandthirdparties/Git-Training/git-training
Branch	master
Topic	
Updated	1 second ago

life.augmented

Lab : Pull Requests

Pull Requests

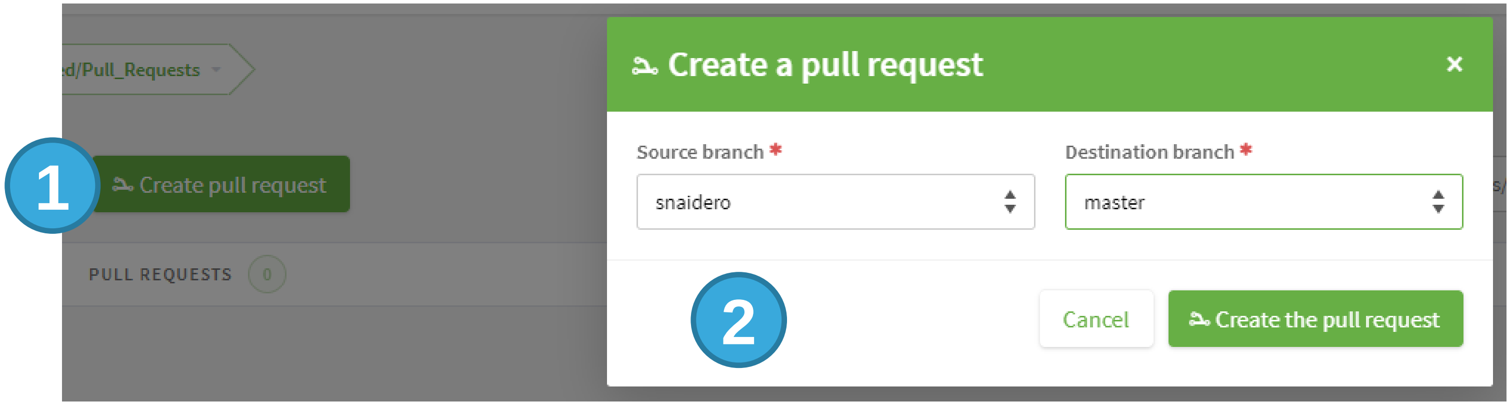
- Clone

  ssh://gitolite@codex.cro.st.com/git-trainings/Advanced/Pull_Requests.git

- Create a feature/<login> branch & do a commit in users/<login>

- Push your branch

- Create your pull request in Codex
  - https://codex.cro.st.com/plugins/git/git-trainings/Advanced/Pull_Requests

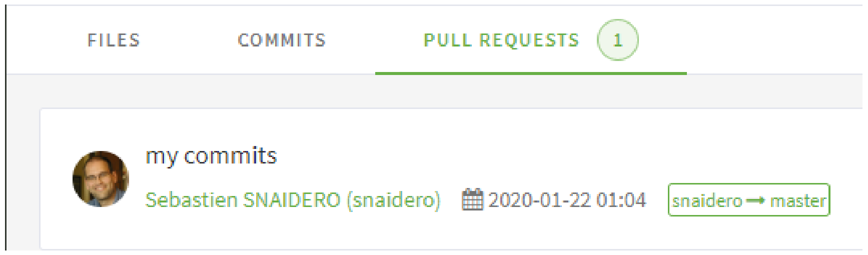


Source branch *: snaidero

Destination branch *: master

Pull Requests

- Explore the Pull Request in the Codex Web page



my commits

Sebastien SNAIDERO (snaidero) 2020-01-22 01:04 snaidero → master

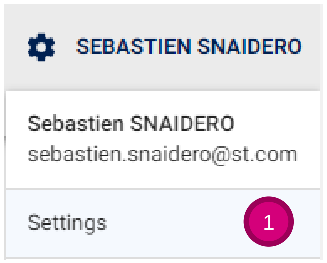
- Go to 'Changes' and insert a comment attached to a line in a file
- Update your code, commit & push
- See changes in pull request
- Require a rebase & a squash through a comment
- Rebase -i your branch & push (you are allowed to rewrite feature/*)
- Finally merge the Pull Request
- Refresh your local repo (fetch) to see changes
- Delete both local & remote feature/<login> branch

Annexes

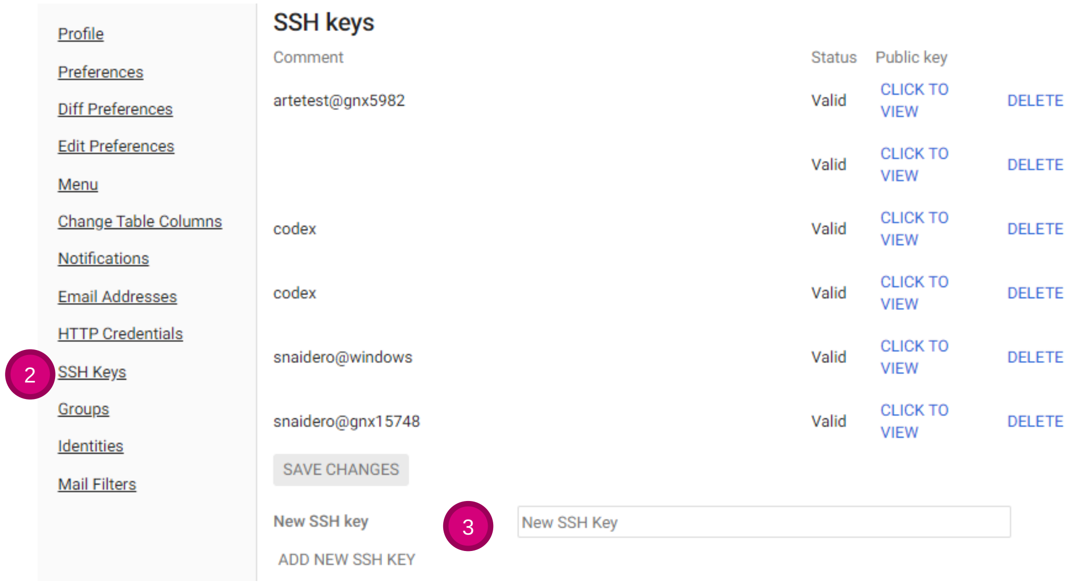
Install ssh keys on windows

- To access your Git repositories you will need to create and install SSH keys

- To do this you need to run git Bash. Open it from your start menu
  - Generate ssh key through the command : ssh-keygen -t rsa
    - Press enter at each command dialog (3 times)
  - Go to folder: C:\Users\username\.ssh
    - Open id_rsa.pub
    - Copy all content



- Go to gerrit website
  - Go to your account setting
  - Click on SSH Publick Keys
    - Click on "Add Key …"
    - Paste id_rsa.pub content there



SSH keys

Comment	Status	Public key	
artetest@gnx5982	Valid	CLICK TO VIEW	DELETE
	Valid	CLICK TO VIEW	DELETE
codex	Valid	CLICK TO VIEW	DELETE
codex	Valid	CLICK TO VIEW	DELETE
snaidero@windows	Valid	CLICK TO VIEW	DELETE
snaidero@gnx15748	Valid	CLICK TO VIEW	DELETE

New SSH key: New SSH Key

life.augmented

Add SSH key to your Gerrit account

- Log into the web interface for Gerrit.
- Click on your username in the top right corner, then choose "Settings".
- Click "SSH Public keys" in the menu on the left.
- Paste your SSH Public Key into the corresponding field.