Gitlab Configuration Details

Version 1.18-SNAPSHOT

2018-09-11

Table of Contents

1. Introduction 2
2. Server configuration 2
3. Project creation 2
    3.1. Configure rebase without merge commit 2
    3.2. Protect master branch 3
    3.3. Configure Eclipse 3
    3.4. Merge Requests workflow 3
    3.5. Pipeline creation 4
4. Appendix 6
    4.1. Revision marks 6

Table 1. History

Date	Author	Detail
2018-09-05	bcouetil	Minor changes
2018-08-29	bcouetil	Asciidoc HTML look & feel changes
2018-08-24	bcouetil	Reworked asciidoc index.html
2018-08-24	bcouetil	Icones added for download + favicon added for webpage
2018-08-23	bcouetil	Initial commit

1. Introduction

Gitlab in his free online plan offers some nice features :

- 2,000 CI pipeline minutes per group per month on our shared runners
- Unlimited private projects and collaborators
  - Built-in CI/CD
  - Cycle Analytics
  - Issue Boards
  - Time tracking
  - Preview your changes with Review Apps
  - Publish static websites for free with GitLab Pages
  - Git LFS 2.0 support

2. Server configuration

TODO

3. Project creation

- Create a project in web interface.
  - check **Create README.md** or else **master** branch is not created
- Import git project in Eclipse
- Change git config, adding :

```
[user]
  name = yourname
  email = youraccount@yourprovider.com
```

3.1. Configure rebase without merge commit

- In the web interface, go to **Project** → **Settings** → **General** → **Merge request**, and configure as follow :
  - ☑ Fast-forward merge
  - ☑ Only allow merge requests to be merged if the pipeline succeeds
  - ☐ Only allow merge requests to be merged if all discussions are resolved

3.2. Protect master branch

Gitlab workflow is not as straightforward as Gerrit's (see below). In Eclipse, it's too easy to push to master instead of our current Merge Request's branch.

Here is how to protect it :

- In Gitlab interface, navigate to your project → **Settings** → **Repository** → expend **Protected Branches** and configure :
  - Branch = **master**
  - Allowed to merge = **Maintainers**
  - Allowed to push = **No one**

3.3. Configure Eclipse

Clone the repository as you would do for any Git repository, see Eclipse Best Practices for details.

3.4. Merge Requests workflow

We assume that we will be the only one to push on that branch, so we do not check-out the branch, and just push there. This helps do fewer interactions in Eclipse.

Initialize the change

- In Gitlab
  - Go to **Issues** → **New issue**
  - Set information and create
  - Click **Create merge request**
- In Eclipse
  - right click on your repo → **Pull**
    - The branch should appear under **Branches** → **Remote Tracking**
  - In **Remotes** → **origin**, right click on the second address and choose **Configure Push...**
  - Click **Add...**

    Remote branch = <type the number to get the full branch>

    ☑ Force update

    The specification should be something like

    ```
    +HEAD:refs/heads/4-minor-asciidoc-changes
    ```

    Press **OK** then **Save**

Eclipse should now be ready to handle smoothly Gitlab workflow. You can check your repository git config which should be like :

git/config

```
[core]
  repositoryformatversion = 0
  filemode = false
  logallrefupdates = true
[remote "origin"]
  url = https://gitlab.com/bcouetil/academy.git
  fetch = +refs/heads/*:refs/remotes/origin/*
  push = +HEAD:refs/heads/4-minor-asciidoc-changes
[branch "master"]
  remote = origin
  merge = refs/heads/master
  rebase = true
[user]
  name = myname
  email = myaccount@myprovider.com
```

dev iterations

- The first time
  - Update your files…
  - Go to **Git Staging** view
  - Stage your files
  - Hit **Commit and Push…** and **Close**
- On each new iteration do the same and

  ☑ toggle ON **Amend**

  change the text if needed

Don’t toggle on **Add Change-Id**, this changes the behavior of **Commit and Push…** towards Gerrit style.

More details and screenshots in Eclipse Best Practices

Now a pipeline should be launched in Gitlab interface.

Merge in Gitlab

Browse the Merge Request in Gitlab

☑ Remove source branch

Pull in Eclipse to be up to date

3.5. Pipeline creation

Pipeline

```
image: maven:latest

stages:
  - build
  - test
  - deploy

variables:
  #MAVEN_CLI_OPTS: "-s cg-settings.xml --batch-mode"
  MAVEN_CLI_OPTS: "--batch-mode"
  MAVEN_OPTS: "-Dmaven.repo.local=.m2/repository"

cache:
  paths:
    - .m2/repository/

build:
  stage: build
  script:
    - apt-get update -qq && apt-get install -qq --assume-yes libc6-i386
    - mvn clean dependency:purge-local-repository
    - mvn install -DskipTests -Dassembly.skipAssembly=true


# the pipeline step must be named 'pages' for gitlab to deploy locally (in addition to github)
pages:
  stage: build
  script:
    #- ./src/scripts/asciidocOnlyModified.sh
    - ./src/scripts/asciidocHistory.sh .
    - apt-get update -qq && apt-get install -qq --assume-yes graphviz
    - git config --global user.email "gitlab@noreply.com"
    - git config --global user.name "GitLab"
    - mvn generate-resources -Dadoc.skip=false --non-recursive
    - cp src/docs/asciidoc/*.adoc target/generated-docs/
    - cp -R src/docs/asciidoc/subdocs target/generated-docs/
    - mvn scm-publish:publish-scm
    - mkdir public
    - mv target/generated-docs/* public/
  artifacts:
    paths:
      - public
  #only:
  #  - master


unit-test:
  stage: test
  script:
    - mvn clean
    - mvn test -Dcheckstyle.skip=true -pl cg-utils

integration-test:
  stage: test
  script:
    - mvn clean
    - mvn verify failsafe:verify -Dcg.ut.skip=true -Dcheckstyle.skip=true -pl cg-utils

assembly:
  stage: deploy
  script:
    - mvn clean
    - mvn install -DskipTests -Dcheckstyle.skip=true -pl cg-utils
  #only:
  #  - master
```

4. Appendix

4.1. Revision marks

Differences since last tag