Travail pratique #3: GitLab

École polytechnique de Montréal

Trimestre : Automne 2019

Équipier1: *William Younanian*
Équipier2: *Vincent Samuel-Lafleur*

Équipe: 411

Présenté à : Annie Rochette

Polytechnique Montréal
Remis le 9 mars 2020

4. Partie 1

4.1 Gestion des bogues

1. Qu'est-ce que le logiciel Inkscape et à quoi sert-il ?

Inkscape est un logiciel de dessin libre d'accès qui utilise le standard W3C.

2. Comment est déterminé le niveau d'importance d'un bogue (*Issue*) ?

L'équipe travaillant sur le bogue décide du niveau d'importance du bogue.

3. Sur la barre latérale gauche du projet, sélectionnez la section Issues puis la colonne All, et entrez dans la barre de recherche l'id suivant : #459 . En quoi consiste le bogue lié à cet id ?

Impossibilité d'ouvrir des documents suite à une mise à jour MacOS en faisant Fichier > Ouvrir.

4. Quel est le nom de l'utilisateur qui a rapporté ce bogue ?

Windell Oskay

5. Quel est le niveau d'importance de ce bogue ?

High

6. Combien de personnes ont participé à la discussion reliée à ce bug ?

Vingt-quatre personnes

7. Combien de merge request sont reliés à ce bogue ? Combien ont été acceptés ?

Deux merge requests ont été faits. Les deux ont été acceptés.

8. Est-ce que la communauté a confirmé la validité du bogue ? Si oui, comment?

Oui, plusieurs personnes ont confirmé la validité du bogue dans les commentaires en disant avoir vécu la même situation que ce qui était initialement décrit. Les réactions de la communauté, en haut des commentaires, montrent aussi un appui à la validité du bogue. De plus, la présence de plusieurs *thumbs up* peuvent donner un indice sur le fait que ce bogue est effectivement valide.

9. Est-ce que le bogue a été résolu ? Qu'est-ce qui vous l'indique ?

Oui, le bogue est fermé avec 9 tâches sur 9 d'accomplies.

10. Refaire les étapes 3 à 9 inclusivement, mais en entrant dans la barre de recherche l'id suivant : ***#952***

3: Une décoration de texte est ignorée lorsqu'un fichier est converti en pdf.

4: Bogue rapporté par Nathan Lee

5: Niveau d'importance : Moyen

6: Six personnes ont participé à la discussion reliée à ce bogue.

7: Aucun merge request n'a été fait.

8: Oui, tout comme pour le bogue précédent, la communauté a confirmé la présence de ce bogue en partageant dans les commentaires leur volonté de le voir être réglé rapidement. Encore une fois, la présence de *thumbs up* peuvent donner un indice supplémentaire sur la validité.

9: Non, le bogue est toujours ouvert.

4.2 Révision technique de code source

Consultez le projet open source suivant : https://gitlab.com/ase.

1. Que représente le projet open source ASE et à quoi sert-il ?

Le projet ASE est une librairie de code Python ayant été développée pour aider à la réalisation de simulations atomiques (Atomic Simulation Environment).

2. Dans un contexte de contribution à un projet, qu'est-ce qu'une révision (patch) ?

Une révision est une série de changement dans le code ayant pour but d'améliorer ou de réparer certains bogues dans un code source.

3. Sur la barre latérale gauche du projet, sélectionnez la section *Merge Requests* puis la colonne *All,* et entrez dans la barre de recherche le nom suivant : *WIP: Data driven calculator.* Décrivez brièvement ce que fait cette révision.

Cette révision ajoute une calculatrice qui prend en charge plusieurs codes.

4. Qui est l'auteur de cette révision ?

Hjorth Larson

5. *WIP signifie Work In Progress. Pourquoi l'auteur du commit a choisi de le préciser ?*

Puisqu'il ne s'agit pas de la version finale de cette révision. En effet, l'auteur mentionne dans un commentaire que plusieurs éléments restent à implémenter pour supporter plus de calculatrices.

6. Combien de fichier(s) a/ont subi de changement ? Le(s) quel(s) ?

13 fichiers :

ase/calculators/datadriven.py
ase/calculators/emt.py
ase/calculators/openmx/reader.py
ase/calculators/openmx/writer.py
ase/calculators/singlepoint.py
ase/io/formats.py
ase/io/openmx.py
ase/test/datadriven/emt.py
ase/test/datadriven/espresso.py
ase/test/datadriven/main.py
ase/test/datadriven/openmx.py
ase/test/fio/oi.py
ase/test/openmx/openmx.py

7. Quel est le statut actuel de la révision ? Que signifie-t-il ?

Open, ce qui signifie que la révision est en cours et non implémentée.

8. Refaire les étapes 3 à 6 inclusivement, mais en entrant dans la barre de recherche le nom suivant : *Extended fileformat info*

3 : Cette révision rassemble l'information au sujet des objets d'un certain type de fichier et inclut un résumé facilement lisible.

4 : Hjorth Larson

5 : Cette révision n'est pas un work in progress

6 :

Neuf fichiers :

ase/cli/info.py
ase/cli/nomadget.py
ase/io/formats.py
ase/io/lammpsdata.py
ase/test/fio/ioformats.py
ase/test/fio/match_magic.py
ase/test/fio/nomad.py
ase/test/fio/oi.py
ase/utils/__init__.py

4.3 Intégration continue

L'objectif est de savoir (en continu) si les changements ("commits") effectués produisent toujours un logiciel fonctionnel qui passent les cas de tests définis.
Consultez le projet open source suivant : https://gitlab.com/fdroid/fdroidclient.

1. Sur la barre latérale gauche du projet, sélectionnez la section *Merge Requests* puis la colonne *All,* et entrez dans la barre de recherche le nom suivant : *Anti-Feature icons.* Décrivez brièvement ce que fait cette révision.

Ajoute les nouveaux icônes de *Anti-Feature.*

2. En cliquant sur l'onglet *Pipelines*, combien de commits passent les tests ? Combien échouent les tests ? Combien ont été annulés ?

Trois pipelines passent les tests, 6 ont échoué les tests et 3 ont été annulés.

3. En cliquant sur l'onglet *Commits*, combien de commits passent le pipeline ?

Un seul.

4. Quel est le rôle des pipelines sur Gitlab ?

Lorsqu'on fait un *Merge Request* sur Gitlab, un *pipeline* est créé. Celui-ci permet de voir si les différences ajoutée sont qualifiées pour être fusionnées dans la branche cible. Ça sert à automatiser davantage le travail collaboratif.

5. Quel est le lien entre les tests et les pipelines sur Gitlab ?

Le pipeline prend le code le plus récent de la branche source et exécute des tests pour voir si le code fonctionne. Les *Jobs* sont ceux qui disent quoi exécuter, soit la compilation de code ou des tests.

6. Pourquoi les pipelines comportant plusieurs étapes (comme https://gitlab.com/uniqx/fdroidclient/pipelines/109834613) sont utiles pour l'intégration continue?

Parce qu'ils servent à automatiser davantage le travail. Dans l'exemple donné, il y a 4 étapes au pipeline. Ceci permet d'exécuter ces étapes tous dans un même pipeline et ainsi, d'automatiser le travail.

5. Deuxième partie

L'objectif de cette seconde partie est de se familiariser avec le processus de contribution à un projet publié sur Gitlab. Tout d'abord, vous aurez à former des groupes de deux équipes. À tour de rôle, chacune des équipes prendra le rôle de l'équipe 1 et l'équipe 2. L'équipe 1 devra implémenter la fonction demandée et faire un Merge Request, alors que l'équipe 2 fera une révision technique de ce Merge Request.

Attention : Les modifications doivent être faites sur **une branche autre que le master.** Les deux équipes

1. Après avoir créé un compte Libre sur Gitlab, vous recevrez une invitation pour un projet de la part de vos chargés. Le reste du du TP3 se fera sur ce projet Gitlab.
2. Clonez le répertoire localement sur votre machine à l'aide de Git et accédez à votre entrepôt dans le terminal. Quelle(s) commande(s) avez-vous utilisée ?

cd Desktop/tp3-411-412

git clone https://gitlab.com/log1000-h20-tp3/tp3-411-412.git

3. Il n'y a pas de Makefile fourni pour la compilation du projet. Vous devez en écrire un et le mettre sur votre entrepôt.

L'équipe 1 :

MODIFICATION DU CODE #1 : Le but des modifications de cette partie est d'implémenter l'opérateur = de la classe Client qui écrase les attributs de l'objet de gauche par les attributs de l'objet passé en paramètre.

Note : Notre équipe (411) a fait la modification #1 du code.

4. Ajoutez vos modifications à l'entrepôt Git. Quelles commandes avez-vous utilisée ?

git branch equipe1

git checkout equipe1

modifications

git push --set-upstream origin equipe1

Faites un commit et un push de vos changements vers le serveur. Retournez sur votre compte Gitlab, **Projects → Your projects** et puis le répertoire correspondant du TP. Sur la barre de navigation de gauche, accédez à la section Merge Requests

1. Qu'est-ce qu'un Merge Request?

Un merge request est une demande de merge une branche source, par exemple, la branche equipe1, avec une branche target, par exemple, la branche master.

2. Quelle est l'utilité d'un Merge Request? Cliquez sur « New merge request », afin de créer un nouveau Merge Request.

Un merge request sert à s'assurer que toutes les équipes collaborant sur le projet acceptent les modifications apportées au master sur le remote.

3. Quelles sont les branches source et destination correspondantes à votre Merge Request ?

La branche source est la branche remote equipe1 et la branche destination est la branche master remote.

L'équipe 2:

Faites une révision technique du Merge Request effectué par l'équipe 1.

1. Quel(s) est/sont les commits associées ?

Note: au début, l'autre équipe a fait un commit: ajouter operator <<. Ils ont fait un merge request. Par contre, il y avait une erreur dans l'affichage, donc ils ont fais un nouveau commit et un nouveau merge request.

Donc, le commit associé au dernier merge request est: corriger l'operateur <<

2. Quel(s) est/sont les fichiers qui ont été modifiés ?

Panier.cpp

3. Est-ce que les modifications demandées sont correctes ?

Oui, les modifications apportées sont correctes.

4. Est-ce qu'il y a des objets (.o) et/ou l'exécutable du projet présents sur le Git? Est-ce que c'est une bonne ou mauvaise pratique de mettre ces fichiers dans un entrepôt Git?

Non, il n'y en a pas. Ce n'est pas une bonne pratique, car ce sont des fichiers inutiles qui prennent de la place pour rien.

5. Attribuez un score aux modifications et justifiez votre choix.

Score de 10/10, car les modifications et l'affichage sont parfaites.

Ajoutez des commentaires, incluant le score et votre justification et publiez-les dans la discussion du Merge Request.

Dans le cas des groupes de deux équipes : Il est maintenant temps aux deux équipes d'inverser les rôles.
MODIFICATION DU CODE #2 : Le but des modifications de cette partie est d'implémenter l'opérateur << (remplace l'affichage) de la classe Panier, qui affiche un panier selon la forme demandée (voir capture d'écran)