

Suivez cette présentation sur votre ordinateur :

<https://louvainlinux.org/atelier-git>

Préparez-vous à utiliser git : vous utiliserez le logiciel GitHub Desktop durant cette présentation.

Prenez un peu d'avance, créez-vous un compte Github et installez Github Desktop :

- Sur les ordinateur Windows UCL : installez <https://desktop.github.com>
- Ou installez GitHub Desktop sur votre ordinateur :
 - **Ubuntu** : <https://github.com/shiftkey/desktop/releases>
 - **Windows ou OS X** : <https://desktop.github.com>



Présentation Git

Un outil de collaboration puissant

Théo Vanden Driessche Morgan Leclerc

10 Octobre 2019

KAP Louvain-li-Nux

- Cette présentation est sous license libre CC-BY 4.0.
- En ligne (slides en pdf et sources \LaTeX , exercices...) :
<https://github.com/louvainlinux/atelier-git>

Table des matières

1. Introduction
2. Principes de Git
3. Utilisation : en pratique
4. Exercices
5. Fonctionnalités plus avancées
6. Informations et ressources

Introduction

Comment gérez-vous actuellement un projet ?

- L'envoyer à travers un message sur Facebook, ... (**Très mauvaise idée**)
- L'envoyer par mail (**Un peu moins**)
- Utiliser une Dropbox, Google Drive, ... (**Déjà mieux mais toujours risqué ou manque de fonctionnalités**)

Solution : Utiliser un **système de gestion de version décentralisé** (Distributed Version Control System (DVCS) pour les anglophiles).

Un DVCS ?

- **Version** Enregistre des « instantanés » du projet.
- **Gestion** Revenir en arrière, voir des différences, fusionner des modifications.
- **Décentralisé** Chacun
 - a sa copie (avec son historique) sur son PC,
 - peut mettre sa copie (et son historique) en ligne,
 - peut récupérer sur son PC les copies et historiques disponibles en ligne,
 - peut fusionner différentes copies (semi-)automatiquement.
- **Projet** n'importe quel répertoire (« dossier ») sur votre ordinateur. Donc n'importe quoi : Bureautique, \LaTeX , code, images, musique...

Et Git dans tout ça ?

Git a été créé en 2005 par Linus Torvalds (auteur de Linux) ; le plus connu et utilisé.

À l'origine, interface en ligne de commande.

Aujourd'hui : aussi des interfaces graphiques, dont GitHub Desktop.

Mais on m'avait parlé de GitHub !

Souvenez-vous...

- **Décentralisé** Chacun

- peut mettre sa copie (et son historique) en ligne,
- ...

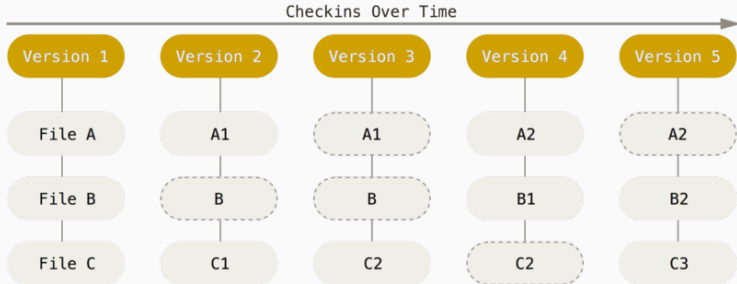
Il y a plein d'"endroits" en ligne où on peut envoyer son travail, GitHub est le plus connu.

En plus de ça, GitHub a des fonctionnalités pour interagir avec des collaborateurs.

Principes de Git

- **Espace de travail** : les fichiers, répertoires... dans lesquels on travaille. Ils n'ont rien de spécial par rapport à d'autres dossiers sur l'ordinateur.
- **Dépôt** : espace de travail + historique, sur un ordinateur.
- **Commit** : "version", est le successeur d'une autre commit.
- **Historique** : la "chaîne" de tous les commits, du plus ancien au plus récent.
- **Dépôt distant** : un dépôt qui se trouve chez GitHub.

Concept : le commit



Les illustrations non-sourcées viennent de <https://git-scm.com/book>.

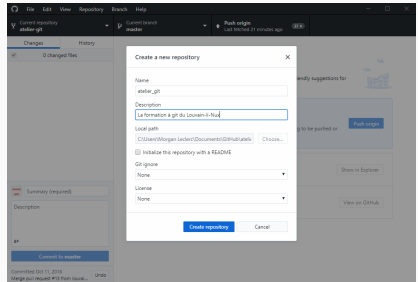
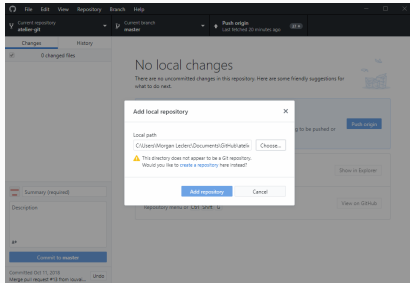
- **Créer** un dépôt sur GitHub.
- **Cloner** (faire une copie d') un dépôt de GitHub sur son PC.
- **Modifier/créer** des fichiers (pas avec Git!).
- **Ajouter** un fichier modifié : il sera pris en compte dans le prochain commit.
- **Faire un commit** : créer une nouvelle version, qui contient tous les fichiers ajoutés. On y ajoute un commentaire (qui décrit les changements).

- **Consulter** un historique.
- **Push** : envoyer ses nouveaux commits sur GitHub.
- **Pull** : récupérer des changements (qui ont été envoyés par quelqu'un d'autre) depuis GitHub.
- **Merge** : quand on Pull et qu'on a aussi des nouveaux commits sur son PC. Git essaye de fusionner automatiquement ; s'il ne sait pas le faire, il demande à l'utilisateur.

Questions ?

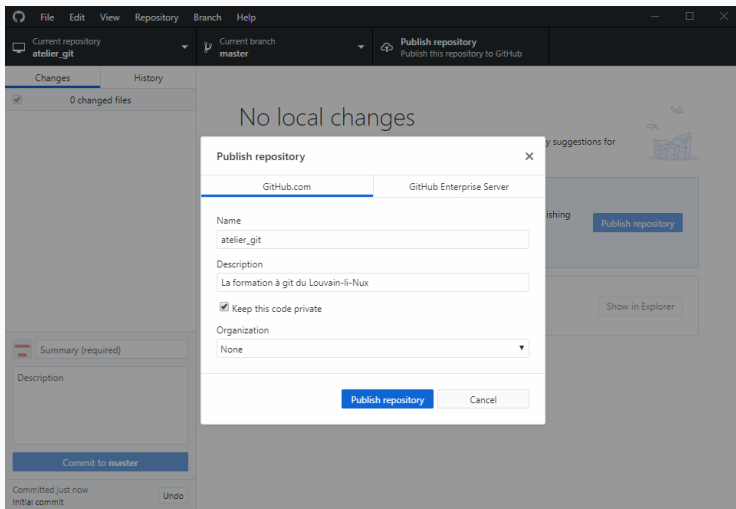
Utilisation : en pratique

Créer un nouveau dépôt local



File → Add Local repository *Ctrl+O*

Publier un dépôt sur Github



Pour accéder au dépôt en ligne : Repository → View on Github
Ctrl+Shift+G

Ajouter un collaborateur sur GitHub

reirep / **Data-Analytics-Applied-in-Business** Private


Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights **Settings**

- Options
- Collaborators**
- Branches
- Webhooks
- Integrations & services
- Deploy keys

Collaborators

Push access to the repository

 Justenult ×

Search by username, full name or email address

You'll only be able to find a GitHub user by their email address if they've chosen to list it publicly. Otherwise, use their username instead.

Add collaborator

Ajouter des fichiers

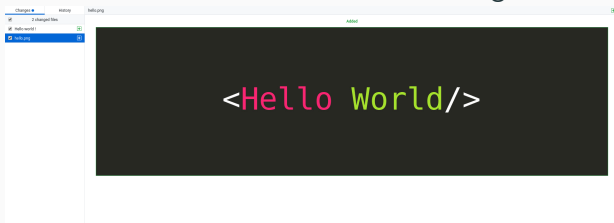
Changes ●		History		Hello world !	
<input checked="" type="checkbox"/>	1 changed file				@@ -0,0 +1,2 @@
<input checked="" type="checkbox"/>	Hello world !			1	+Bonjour tout le monde !
				2	+Comment allez vous ?

Remarque : fichier texte vs binaire

- Fichiers texte : programme, \LaTeX ...

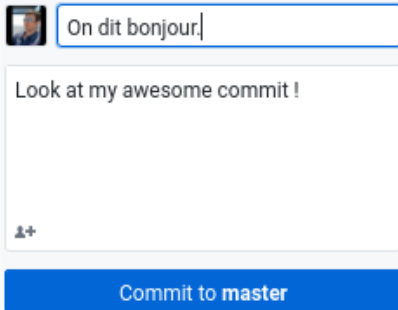
Changes	History	Hello world !
<input checked="" type="checkbox"/> 1 changed file		<code>@@ -0,0 +1,2 @@</code>
<input checked="" type="checkbox"/> Hello world !	<input data-bbox="614 298 635 319" type="button" value="+"/>	<div><div>1</div><div>+Bonjour tout le monde !</div></div> <div><div>2</div><div>+Comment allez vous ?</div></div>

- Fichiers binaires : le reste : Word, Writer, images, sons, PDF...



Créer un commit

- Créer un commit sur base des fichiers ajoutés.
- Message de commit : décrit les changements effectués.



A screenshot of a web-based commit creation interface. At the top left is a small square profile picture of a man. To its right is a text input field with a blue border containing the text "On dit bonjour." with a cursor at the end. Below these is a larger text area with a light blue border containing the text "Look at my awesome commit !". In the bottom left corner of this text area is a small icon of two people with a plus sign. At the bottom of the interface is a solid blue button with the text "Commit to master" in white.

Visualiser l'historique

Changes

History

Merge pull request #279 from Expensi...

Cole committed 14 hours ago

Don't reply to pseudo-sockets

Tyler Karaszewski committed 15 hour...

Merge pull request #278 from Expensi...

Cole committed 2 days ago

Missed instance of alarmDuration

Tyler Karaszewski committed 2 days a...

Merge pull request #277 from Expensi...

Cole committed 2 days ago

give up on HTTPS requests if shutdown...

Tyler Karaszewski committed 2 days a...

Only include each crash command onc...

Tyler Karaszewski committed 2 days a...

Merge pull request #275 from Expensi...

Cole committed 5 days ago

Merge pull request #276 from Expensi...

Cole committed 6 days ago

Enable mutex timeout in Makefile

Tyler Karaszewski committed 6 days a...

Adds logging suggested by sqllite

Tyler Karaszewski committed 6 days a...

Merge pull request #274 from Expensi...

Cole committed 7 days ago

Continue as soon as we see a comma...

Tyler Karaszewski committed 7 days a...

Add logging to determine if slow time...

Tyler Karaszewski committed 7 days a...

Merge pull request #273 from Expensi...

Tyler Karaszewski committed 7 days a...

Mark as complete when we move lists

Cole Eason committed Feb 15, 2018

Mark timed out HTTPS requests as co...

Cole Eason committed Feb 15, 2018

Merge pull request #272 from Expensi...

Cole committed Feb 14, 2018

Missed instance of alarmDuration

Tyler Karaszewski committed • 1bb919 1 changed file

libstuff/libstuff.h

@@ -265,7 +265,7 @@ struct SStopwatch {

265 265

266 266 // Accessors

267 267 uint64_t elapsed() { return StimeNow() - starttime.load(); }

268 267 - uint64_t ringing() { return alarmDuration && (elapsed() > alarmDuration.load()); }

268 + uint64_t ringing() { return alarmDuration.load() && (elapsed() > alarmDuration.load()); }

269 269

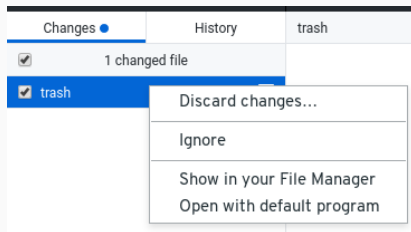
270 270 // Mutators

271 271 void start() { starttime.store(StimeNow()); }

Astuce : ignorer des fichiers

Des fichiers que vous ne voulez jamais dans Git (résultats de compilation, fichiers temporaires...) ? Cachez-les !

NB : Cela crée un fichier `.gitignore` : celui-là, on le versionne.



Push : envoyer des commits sur GitHub

The screenshot shows the Bedrock IDE interface. At the top, there's a menu bar with 'File', 'Edit', 'View', 'Repository', 'Branch', and 'Help'. Below the menu bar, there's a toolbar with 'Current repository' (Bedrock), 'Current branch' (master), and 'Push origin' (Last fetched 3 minutes ago). The main area is divided into two panes. The left pane shows the 'History' tab with a list of commits: 'Yolo' (Pierre Ortegat committed just now), 'Merge pull request #379 from Expensi...' (Cole committed 15 hours ago), 'Don't reply to pseudo-sockets' (Tyler Karaszewski committed 16 hours ago), 'Merge pull request #378 from Expensi...' (Cole committed 2 days ago), and 'Mixed instances of alarmDuration'. The right pane shows the diff for the commit 'Don't reply to pseudo-sockets' by Tyler Karaszewski (commit 3cb967f). The diff shows changes in 'BedrockServer.cpp'. The changes are highlighted in green and blue. The diff shows the following code changes:

```
@@ -1367,12 +1367,17 @@ void BedrockServer::
1367      void BedrockServer::_reply(BedrockCommand& c
1368      SAUTOLOCK(_socketIDMutex);
1369
1370 + // Finalize timing info even for command
1371 + command.finalizeTimingInfo();
1372 +
1373 + // Don't reply to commands with pseudo-c
1374 + if (command.initiationClientID < 0) {
```

Pull : récupérer des commits qui sont sur GitHub

The screenshot displays the GitHub pull request interface for repository **Bedrock**. The current branch is **master**, and the pull origin is **Expensify/tyler-increase-worker-wait-tim...**, last fetched 4 minutes ago. The pull request is titled **Merge pull request #369 from Expensify/tyler-increase-worker-wait-tim...** and was committed by Cole (9f642a5) with 1 changed file.

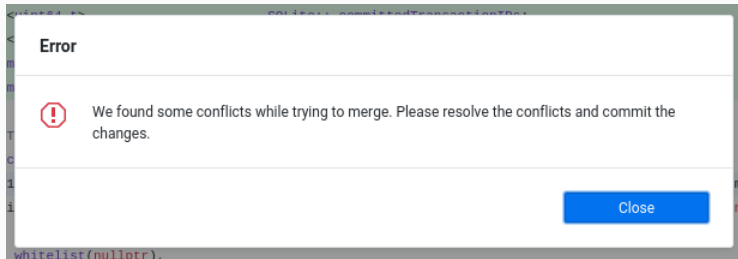
The commit message is **use fixed-decimal ms timing** by Tyler Karaszewski, committed Feb 13, 2018. The description of the pull request is **Add logging, change timeout to 10ms - reviewed, merging**.

The diff shows changes to **BedrockServer.cpp**:

File	Line	Original	Modified
BedrockServer.cpp	4		#include "BedrockPlugin.h"
BedrockServer.cpp	5		#include "BedrockConflictMetrics.h"
BedrockServer.cpp	6		#include "BedrockCore.h"
BedrockServer.cpp	7		+#include <iomanip>
BedrockServer.cpp	8		
BedrockServer.cpp	9		set<string>BedrockServer::_blacklistedParallelCommands;

Merge non-automatique : quand il y a des conflits

Message d'erreur :



Merge non-automatique : quand il y a des conflits

Trouver le(s) fichier(s) en conflit :

Bedrock		
Changes ●		History
✓	18 changed files	
✓	BedrockConflictMetrics.cpp	●
✓	BedrockServer.cpp	●
✓	BedrockServer.h	●
✓	libstuff/libstuff.cpp	●
✓	libstuff/libstuff.h	⚠
✓	libstuff/SHTTPSManager.cpp	●
✓	libstuff/SHTTPSManager.h	●
✓	libstuff/sqlite3.c	●
✓	libstuff/sqlite3.h	●
✓	Makefile	●
✓	sqlitecluster/SQLite.cpp	●
✓	sqlitecluster/SQLite.h	●
✓	sqlitecluster/SQLiteNode.cpp	●
✓	sqlitecluster/SQLiteNode.h	●
✓	test/cluster/test/.../TestPlugin.cpp	●
✓	test/cluster/test/.../TestPlugin.h	●
✓	test/cluster/.../a_MasteringTest.cpp	●
✓	test/tests/SQLiteNodeTest.cpp	●

Merge non-automatique : quand il y a des conflits

Trouver le(s) endroit(s) en conflit dans le fichier (reconnaissables par des balises) :

Avant :

```
// Accessors
<<<<<<<< HEAD
uint64_t elapsed() { return STimeNow() - startTime.load(); }
uint64_t ringing() { return alarmDuration.unload() && (elapsed() > alarmDuration.load()); }
=====
uint64_t elapsed() { return STimeNow() - startTime.load(); }
uint64_t ringing() { return alarmDuration.load() && (elapsed() > alarmDuration.load()); }
>>>>>>>> 3cb967f1fc1b9421cad2f33dce81f258cdeea2b5
```

Après :

```
// Accessors
uint64_t elapsed() { return STimeNow() - startTime.load(); }
uint64_t ringing() { return alarmDuration.load() && (elapsed() > alarmDuration.load()); }
```

Choisir la version que l'on veut garder et commit :



On peut trouver de l'aide :

Github help :

<https://help.github.com/>

Github desktop help :

<https://help.github.com/desktop/>

Exercices

Exercice 1 : partie 1

Exercice à faire par **groupe de 2 ou 3** :

Pour tout le monde :

- Créez un compte sur [GitHub](#).

Une personne du groupe :

- Créez un dépôt nommé *blagues* sur votre compte GitHub. Cocher la case "Initialize this repository...".
- Ajoutez les autres en collaborateurs sur le dépôt. Ils reçoivent une invitation par mail, ils doivent l'accepter.¹

1. Si vous arrivez sur une page 404, connectez-vous à github et ré-essayez.

Exercice 1 : partie 1

Pour tout le monde :

- Installez GitHub Desktop, (voir liens slide 1), puis ouvrez-le.
- Clonez le dépôt sur votre ordinateur.

Exercice 1 : partie 2

Une personne :

1. Créez un fichier avec NotePad++ (ou autre) dans le dépôt sur votre ordinateur (pas besoin d'utiliser GitHub Desktop pour ça). Emplacement par défaut sur les PCs UCL :
Z : \GitHub\blagues.
2. Remplissez le fichier avec des blagues. (Si vous n'avez pas d'idées, cliquez [ici](#))
3. Sauvez le fichier .txt → **NE PAS OUBLIER !!!**
4. Reprenez GitHub Desktop et ajoutez le fichier.
5. Faites un commit.
6. Faites un push vers le dépôt GitHub en ligne.

Exercice 1 : partie 2

Ensuite, les autres, chacun à son tour :

- Faites un pull
- Regardez l'historique pour vérifier que les changements sont là.
- Ajoutez des blagues dans le fichier.
- Refaites les étapes 3 à 6 ci-dessus.

N'hésitez pas à répéter cela plusieurs fois, pour être sûrs de bien comprendre !

Exercice 2 : partie individuelle

Partie à faire individuellement (chacun sur son ordinateur, simultanément) :

- Ouvrez GitHub Desktop et clonez le dépôt précédemment créé si ce n'est déjà fait.
- Ajoutez une blague dans le fichier .txt. Ne pas oublier de sauver le fichier.
- Ajoutez le fichier dans GitHub Desktop.
- Faites un commit. (pas de push)
- Observez et comparez les historiques de chacun.
- Suivez les slides suivants en fonction du nombre personnes dans votre groupe.

Exercice 2 : pour groupe de 2

- Une personne fait un push. L'autre personne ne fait rien.
- L'autre personne fait le pull sur son ordinateur (cliquez sur "push" si "pull" n'est pas affiché, et cliquez "close" sur le message d'erreur qui s'affiche).
- Résolvez le conflit de merge ensemble pour avoir les deux blagues (il faut éditer le fichier en question).
- Une fois le merge terminé, faites un commit, puis un push.
- L'autre personne peut faire un pull pour récupérer la dernière blague.
- Comparez à nouveau vos historiques.

Exercice 2 : pour groupe de 3

- Une personne fait un push. Les autres personnes ne font rien.
- Une des 2 autres personnes fait le pull sur son ordinateur (cliquez sur "push" si "pull" n'est pas affiché, et cliquez "close" sur le message d'erreur qui s'affiche).
- Résolvez le conflit de merge ensemble pour avoir les deux blagues (il faut éditer le fichier).
- Une fois le merge terminé, faites un commit et un push.
- La dernière personne fait le pull sur son ordinateur.
- Résolvez le conflit de merge ensemble pour avoir toutes les blagues.
- Les autres peuvent faire un pull pour récupérer toutes les blagues.
- Comparez à nouveau vos historiques.

Exercice 3

Exercice à faire par **groupe de 2 ou 3** :

Pour une personne :

- Créez un fichier avec Word (.docx) dans le dépôt "*blagues*" précédemment créé sur votre ordinateur.
- Remplir le fichier avec des blagues.
- Sauver le fichier .docx.
- Reprendre GitHub Desktop et faire le add du fichier et le commit.
- Faites un push sur le dépôt GitHub en ligne.

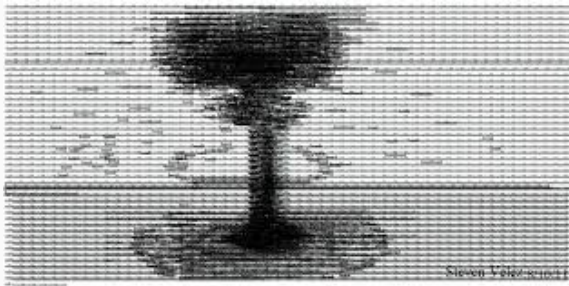
Exercice 3 : partie seul

Pour chacun :

- Faire le pull du dépôt sur votre ordinateur.
- Ajouter une blague au fichier .docx.
- Sauver le fichier .docx
- Faites un commit.

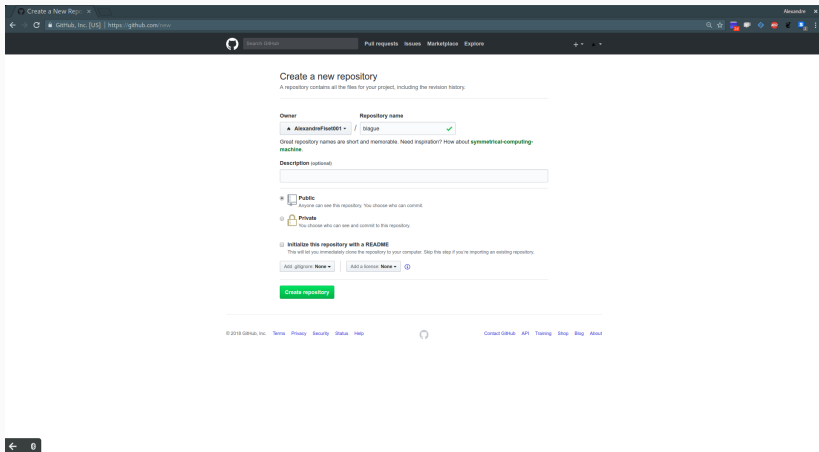
Exercice 3 : partie fun

- Faire les étapes du deuxième slide de l'exercice deux.
- Enjoy :).
- N'hésitez pas a demander pour savoir ce qu'il c'est passé.
#ViveLaTeX!



The screenshot shows the GitHub homepage. At the top, there's a navigation bar with the GitHub logo, search bar, and links for Pull requests, Issues, Marketplace, and Explore. The main content area features a large green banner with the text "Learn Git and GitHub without any code!" and a button to "Start a project". Below this, there's a section for "Repositories you contribute to" and a "New repository" button highlighted with a red circle. The page also displays a list of repositories, including "RailsApps/rails-composer" and "charliegerl/charliegerl/Synthesizer".

Exercise 1 : solutions



Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: AlexandreFournier / Repository name: blagun ✓

Great repository names are short and memorable. Need inspiration? How about [systemical-computing-machine](#).

Description (optional)

☐ Public
Anyone can see this repository. You choose who can commit.

☒ Private
You choose who can see and commit to this repository.

☒ Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

[Add .gitignore](#) [None](#) [Add a license](#) [None](#) ⓘ

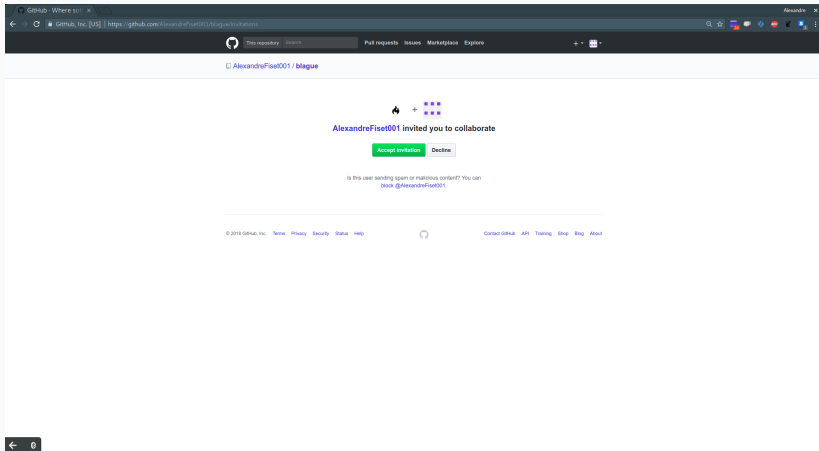
[Create repository](#)

© 2018 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Help](#) [Contact GitHub](#) [API](#) [Training](#) [Shop](#) [Blog](#) [About](#)

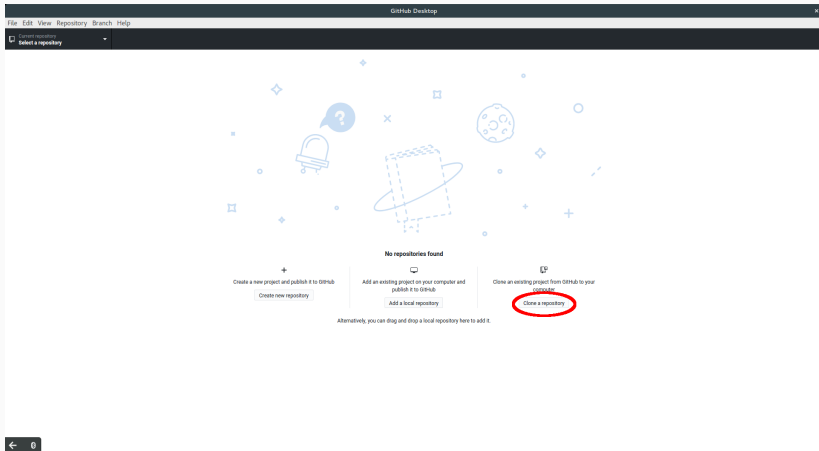
Exercise 1 : solutions

The screenshot shows the GitHub web interface for the repository 'AlexandreFise001 / blague'. The 'Settings' tab is selected in the top navigation bar, marked with a red circle and the number 1. On the left sidebar, the 'Collaborators' link is highlighted with a red circle and the number 2. The main content area shows the 'Collaborators' settings. A red circle and the number 3 highlight the 'Add collaborator' button at the bottom of the search box. Another red circle and the number 4 highlight the 'Copy invite link' button. The page also shows a status message 'Awaiting last1307's response' and a 'Push access to the repository' section.

Exercise 1 : solutions



Exercise 1 : solutions



Exercise 1 : solutions

AlexandreFiset001 / blogue

Watch Star Fork

Code Issues Pull requests Projects Wiki Insights Settings

Quick setup — if you've done this kind of thing before

or <https://github.com/AlexandreFiset001/blogue.git>

We recommend every repository include a [README](#), [LICENSE](#), and [gitignore](#).

...or create a new repository on the command line

```
echo "# blogue" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/AlexandreFiset001/blogue.git
git push -u origin master
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/AlexandreFiset001/blogue.git
git push -u origin master
```

...or import code from another repository

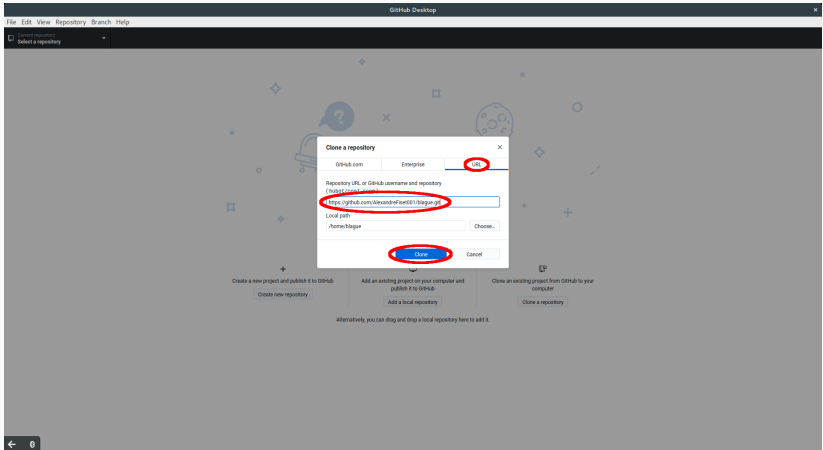
You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

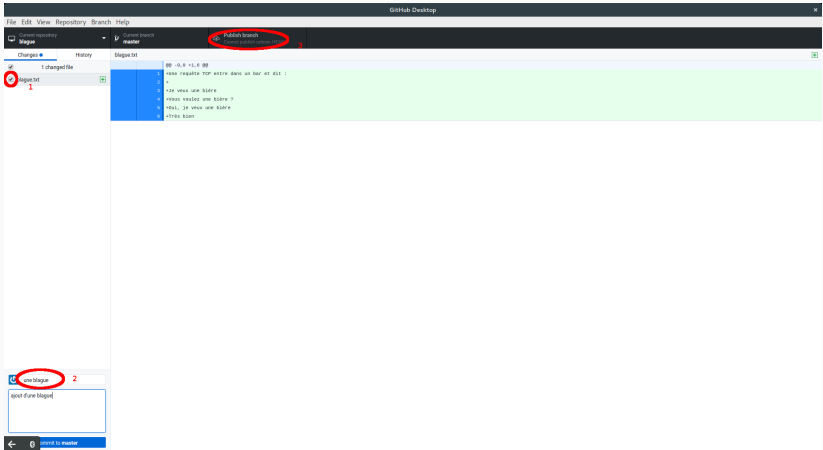
ProTip! Use the URL for this page when adding GitHub as a remote.

© 2018 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Help](#) [Contact GitHub](#) [API](#) [Training](#) [Shop](#) [Blog](#) [About](#)

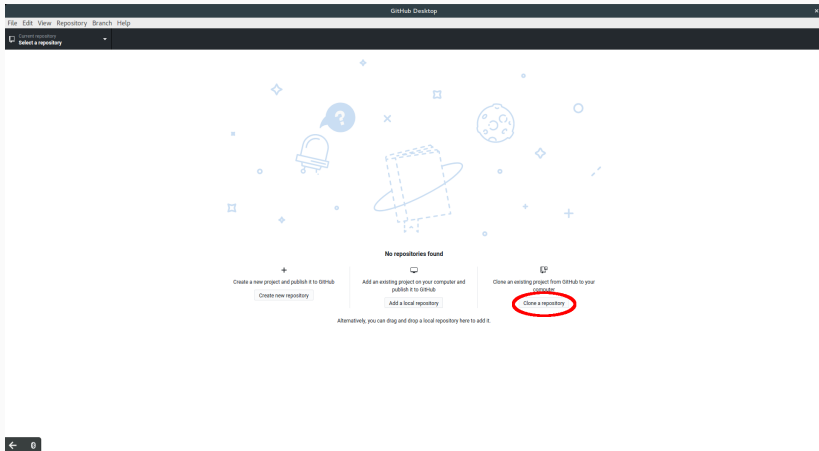
Exercise 1 : solutions



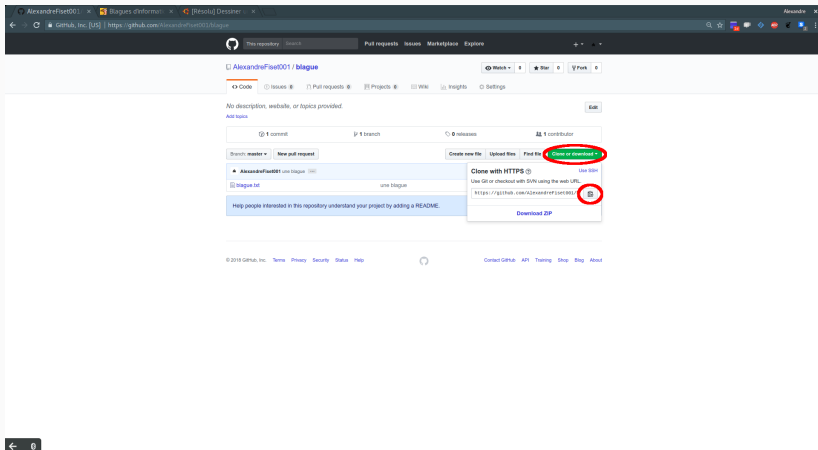
Exercice 1 : solutions



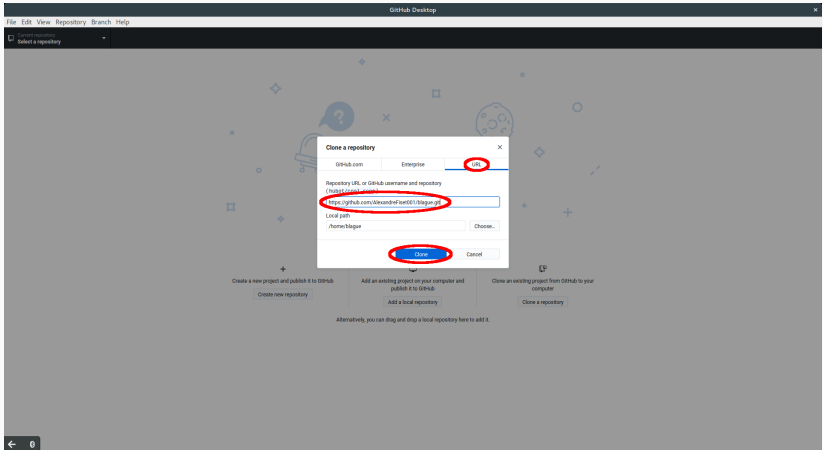
Exercise 2 : solutions



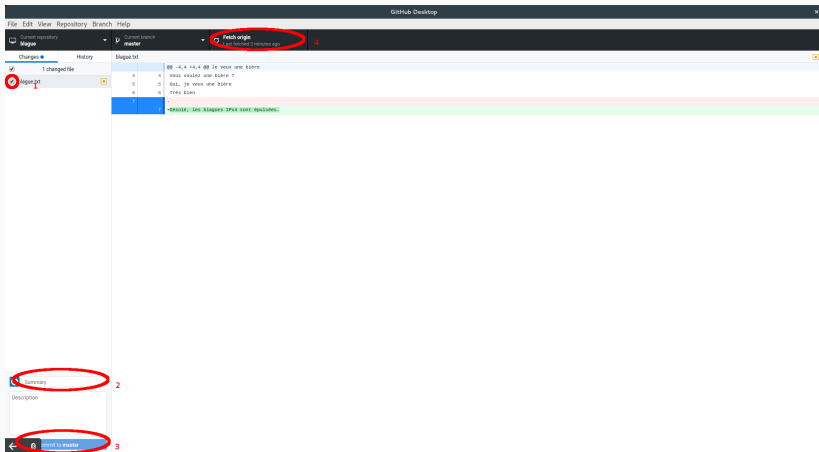
Exercise 2 : solutions



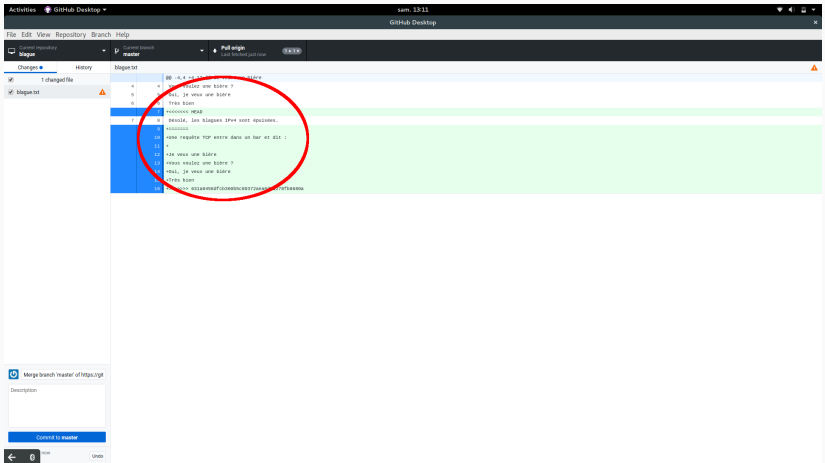
Exercise 2 : solutions



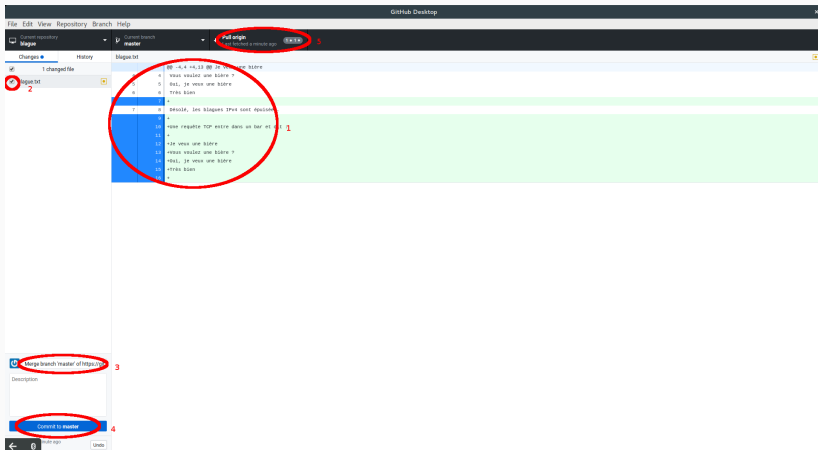
Exercice 2 : solutions



Exercice 2 : solutions



Exercice 2 : solutions



Après le merge, l'historique observé dans GitHub Desktop peut ne pas être le même chez les différentes personnes. Cela est dû au fait que GitHub Desktop n'affiche pas l'entièreté de l'historique, mais seulement les commits qu'il juge pertinents.

Pour voir tout l'historique, aller sur le site github.com ou bien utiliser une autre interface de git (ex. : commande `git log`).

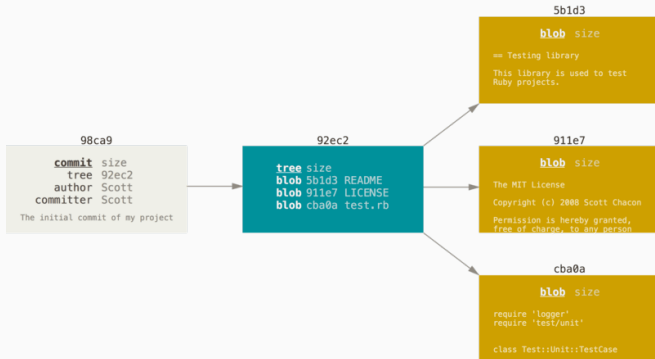
Exercice 3 : solutions

Cette solution est presque identique à la solution de l'exercice 1, sauf qu'il est impossible de faire le merge avec GitHub Desktop. Il faut utiliser une autre interface de git (voir plus loin dans les slides).

Fonctionnalités plus avancées

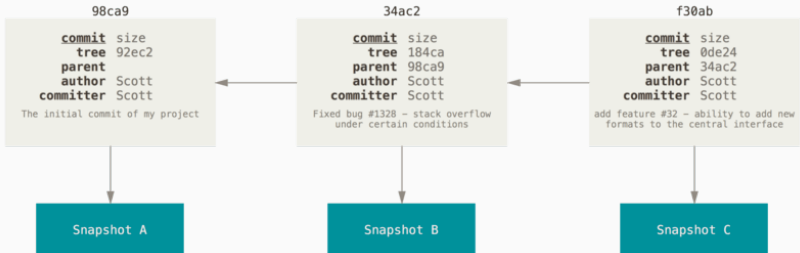
De derrière : les objets git

- Chaque commit a un identifiant :
12f87b95caff8cbeb5ce0717528d77e27db5669c.



De derrière : les parents

- Chaque commit a un parent.



Récupérer un fichier d'un commit passé

Rock solid distributed database specializing in active/active automatic failover and WAN

📄 1,363 commits

🔗 162 branches

📦 3 releases

Branch: master ▼

New pull request

Cre...



coleaeason Merge pull request #382 from Expensify/tyler-checkpoint ...

📁 configs

Use default workerThreads values

📁 docs

Merge with master

📁 libstuff

Missed instance of alarmDuration



















📁 mbedtls @ c49b808

reset mbedtls

📁 plugins

Remove unnecessary warn in MySQL plugin

Récupérer un fichier d'un commit passé


Minor simplification  tylerkaraszewski committed 27 days ago ✖	 c62657e	
Add metrics to see if main thread is overlaoded.  tylerkaraszewski committed 27 days ago ✔	 feba803	
Mitigate timing insssues that cause intermittent failures  tylerkaraszewski committed 27 days ago ✔	 5d8e9c1	
reset mbedtls  tylerkaraszewski committed 27 days ago ✔	 21214ef	
Only log newly queueud commands.  tylerkaraszewski committed 27 days ago ✔	 4826d43	
Add queue size logging on enqueue  tylerkaraszewski committed 27 days ago ✔	 7a90609	

Browse the repository at this point in the history
















Récupérer un fichier d'un commit passé

1,295 commits 162 branches 3 releases 27 contributors LGPL-3.0

Tree: 4826d43ce8 New pull request Create new file Upload files Find file Clone or download

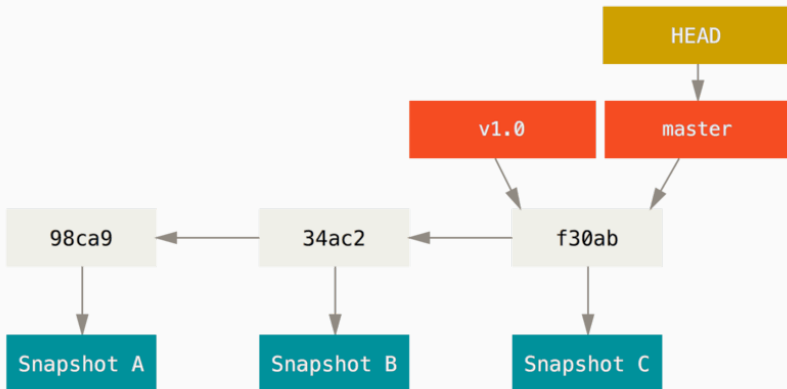
 **tylerkaraszewski** Only log newly queueud commands.

Latest commit 4826d43 27 days ago

 configs	moving the init script to the proper location	a year ago
 docs	Merge with master	6 months ago
 libstuff	Add logging of DNS resolutions	a month ago
 mbedtls @ 4f09291	Only log newly queueud commands.	27 days ago
 plugins	Adding http tests for commands	2 months ago
 sqlitecluster	Commit entire transaction on replicate/synchronize	a month ago
 test	Make mutex a static member	a month ago
 .clang-format	First pass (broken)	a year ago
 .gitignore	Ignore .DS_Store	2 months ago
 .gitmodules	Use HTTPS so no github username required for submodule.	a year ago
 .travis.yml	Revert previous change, travis does not support non-test repo	a year ago
 BedrockCommand.cpp	Add queue size logging on enqueue	27 days ago
 BedrockCommand.h	Clarifying comments/var names.	4 months ago
 BedrockCommandQueue.cpp	Only log newly queueud commands.	27 days ago
 BedrockCommandQueue.h	Only log newly queueud commands.	27 days ago

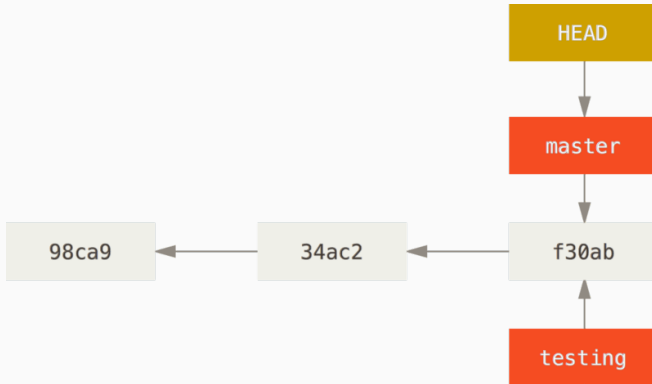
De derrière : les étiquettes

- On peut mettre des étiquettes sur des commits.
- HEAD est la position actuelle.



Créer une branche

- Une branche est une nouvelle étiquette.
- La branche par défaut est master.



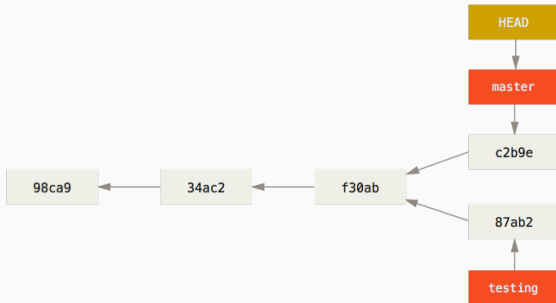
Changer de branche

La branche courante est celle qui suit les nouveaux commits.

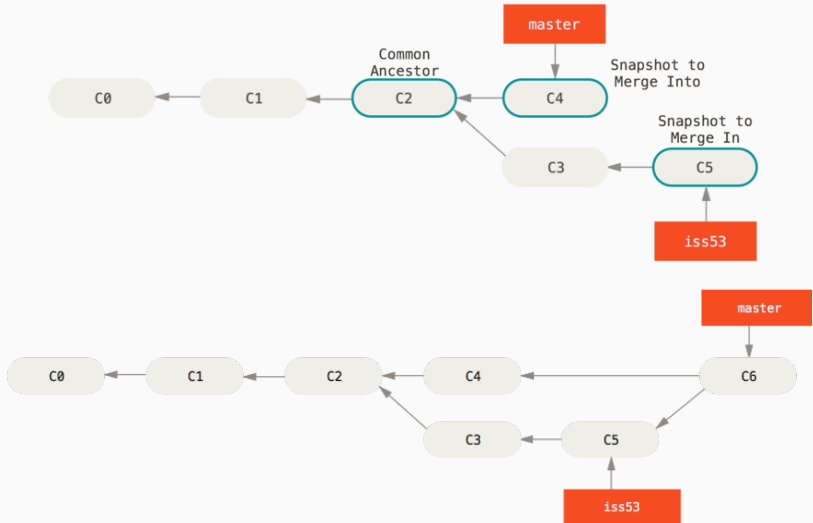


Branches divergentes

- Utilité : travailler sur des modifications indépendantes.



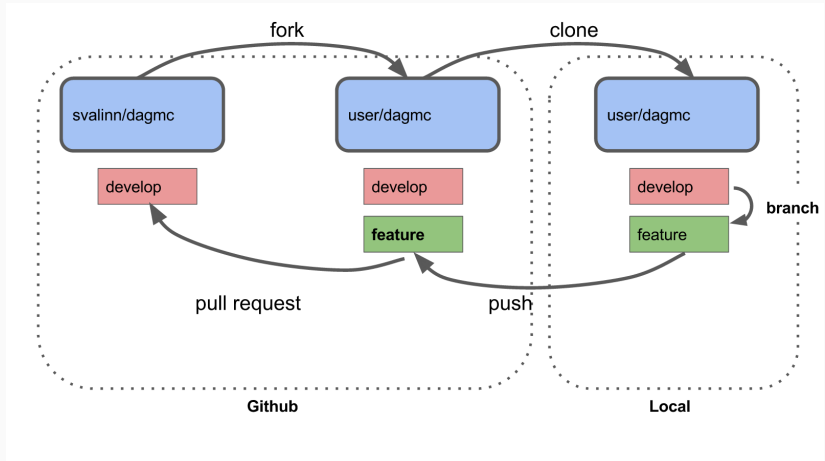
Fusionner des modifications



Parfois il faut résoudre des conflits...

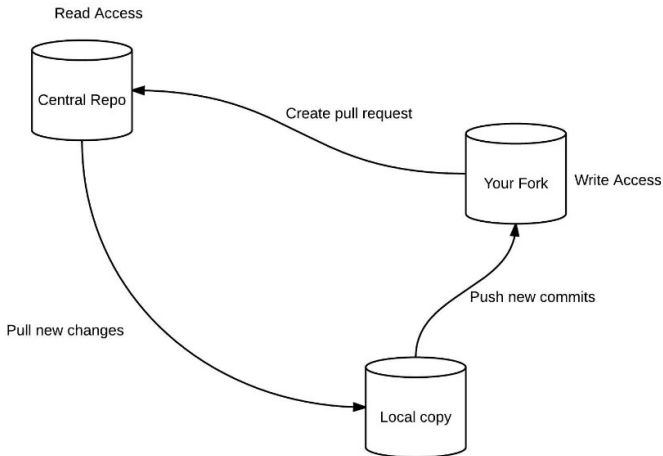
Fork – Pull Request

Une autre méthode de collaboration, très utilisée pour des larges projets et/ou projets où la contribution est ouverte à tous.



Fork – Pull Request : Méthode de travail

Voir <https://help.github.com/articles/fork-a-repo/>.



Forker un dépôt sur GitHub



mozilla-mobile / **focus-android**

Watch 77 Star 633 Fork 223

Code Issues 397 Pull requests 16 Projects 1 Wiki Insights

Firefox Focus: The privacy browser - Browse like no one's watching.

browser android privacy mozilla

1,384 commits 9 branches 34 releases 43 contributors MPL-2.0

Informations et ressources



Pratiquement identiques (tous fonctionnent avec GitHub Desktop).

Dépôts privés gratuits (tout comme sur Gitlab & Bitbucket), et d'autres avantages pour les informaticiens :

<https://education.github.com/pack>.

Nécessite d'ajouter l'adresse ...@student.uclouvain.be au compte GitHub.

Interface en ligne de commande

Utilisée par beaucoup de gens, très puissante si vous êtes à l'aise avec un terminal.

Installation :

- **Ubuntu** : `sudo apt-get install git`
- **OS X** : <https://sourceforge.net/projects/git-osx-installer/>
- **Windows** : <https://git-for-windows.github.io/> (déjà installé à l'UCL)

Documentation :

- **La référence : Git book** : <https://git-scm.com/book> : abordable, bien expliqué et très complet !
- `git help`, `git <command> help`

- <https://git-scm.com/docs/gitk> (Installé par défaut sur PC UCL)
- <https://www.gitkraken.com/>
- D'autres : <https://git-scm.com/downloads/guis>

Questions ?