

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, installez-le :

- 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

Gaëtan Cassiers Alexandre Fiset Pierre Ortegat

1^{er} Mars 2018

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. Installation et configuration
5. Exercices
6. Fonctionnalités plus avancées
7. 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 »). 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,
 - ...

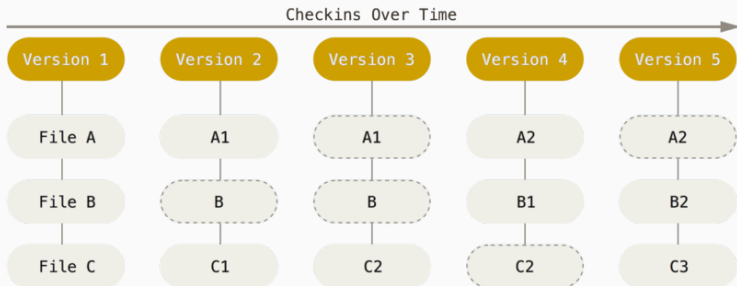
Il y a plein d'"endroits" en ligne où on peut envoyer son travail, GitHub est le plus connu.

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 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 anciens.
- **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 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 de GitHub (qui ont été envoyés par quelqu'un d'autre).
- Merge : quand on Pull et qu'on a aussi des nouveaux commits sur son PC. Git essaye de fusionner automatiquement; s'il ne sais pas le faire, il demande.

Questions?

Utilisation : en pratique

Créer un dépôt sur GitHub

Create a new repository

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

Owner



louvainlinux ▾

Repository name

Tux a l'aventure



Great repository names are **Tux-a-l-aventure** **reimagined-journey**.

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

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

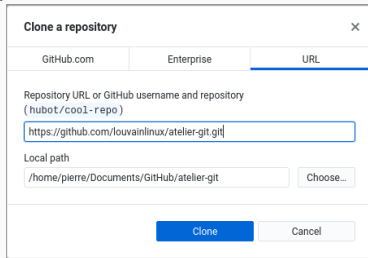
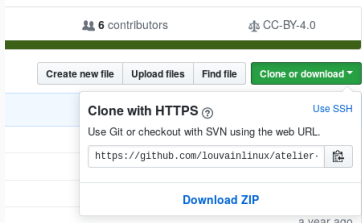
You'll only be able to find a GitHub user by their email address if they've chosen to list it publicly. Otherwise, use their username instead.

Add collaborator

Cloner un dépôt sur son PC

Deux étapes :




1. Prendre l'url du dépôt sur Github
2. Donner l'url a Github desktop



Pour ouvrir cette fenêtre :
File → Clone repository


Ne pas effacer le ".git"!

Ajouter des fichiers

Changes ●		History	Hello world !	
	1 changed file			@@ -0,0 +1,2 @@
	Hello world !		1	+Bonjour tout le monde !
			2	+Comment allez vous ?

Remarque : fichier texte vs binaire

- Fichiers texte : programme, \LaTeX ...

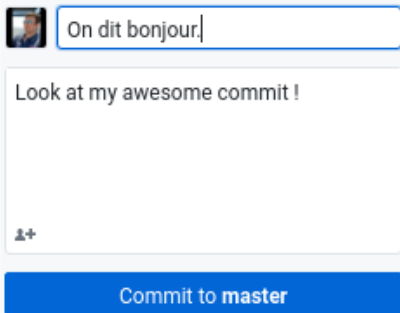
Changes	History	Hello world !
<input checked="" type="checkbox"/> 1 changed file		<code>00 -0,0 +1,2 00</code>
<input checked="" type="checkbox"/> Hello world ! 		<div>1 +Bonjour tout le monde !</div> <div>2 +Comment allez vous ?</div>

- 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 commit creation interface. At the top left is a small profile picture of a person. To its right is a text input field containing the text "On dit bonjour." with a cursor at the end. Below this is a larger text area containing the text "Look at my awesome commit !". At the bottom left of the text area is a small icon of a person with a plus sign. At the bottom of the entire form is a blue button with the text "Commit to master".

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 #378 from Expensi...

Cole committed 2 days ago

Missed instance of alarmDuration

Tyler Karaszewski committed 2 days a...

Merge pull request #377 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 #375 from Expensi...

Cole committed 5 days ago

Merge pull request #376 from Expensi...

Cole committed 6 days ago

Enable mutex timeout in Makefile

Tyler Karaszewski committed 6 days a...

Adds logging suggested by sqite

Tyler Karaszewski committed 6 days a...

Merge pull request #374 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 #373 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 #372 from Expensi...

Cole committed Feb 14, 2018

Missed instance of alarmDuration

Tyler Karaszewski committed 1fb919 1 changed file

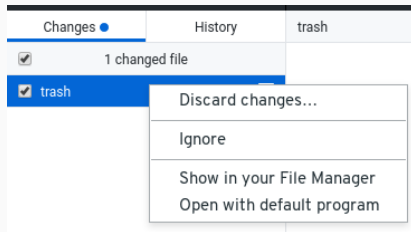
libstuff/libstuff.h

265	265	@@ -265,7 +265,7 @@ struct SStopwatch {
266	266	// Accessors
267	267	uint64_t elapsed() { return STimeNow() - startTime.load(); }
268	268	- uint64_t ringing() { return alarmDuration && (elapsed() > alarmDuration.load()); }
269	269	+ uint64_t ringing() { return alarmDuration.load() && (elapsed() > alarmDuration.load()); }
270	270	// Mutators
271	271	void start() { startTime.store(STimeNow()); }

Astuce : ignorer des fichiers

Des fichiers qu'on ne veut 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), and 'Merge pull request #378 from Expensi...' (Cole committed 2 days ago). The right pane shows the diff for the commit 'Don't reply to pseudo-sockets' by Tyler Karaszewski. The diff shows changes to 'BedrockServer.cpp'. The changes are highlighted in green, indicating additions. The diff shows lines 1367 to 1374, with line 1370 being a new line added. The code in the diff is as follows:

```
@@ -1367,12 +1367,17 @@ void BedrockServer::I
void BedrockServer::_reply(BedrockCommand& c
SAUTOLOCK(_socketIDMutex);

// Finalize timing info even for command
command.finalizeTimingInfo();

// Don't reply to commands with pseudo-
if (command.initiatingClientID < 0) {
```


Pull : récupérer des commits qui sont sur GitHub

File Edit View Repository Branch Help

Current repository: **Bedrock** | Current branch: **master** | Pull origin (Last fetched 4 minutes ago) (25 +)

Changes | **History**

Merge pull request #369 from Expensify/tyler-increase-worker-wait-tim...

Cole committed 9f642a5 1 changed file

..**e-limit**

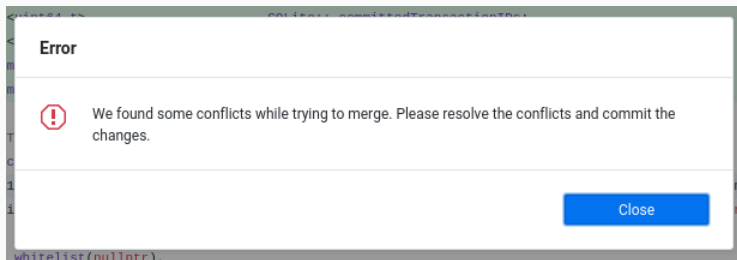
Add logging, change timeout to 10ms - reviewed, merging

BedrockServer.cpp

		@@ -4,6 +4,7 @@
4	4	#include "BedrockPlugin.h"
5	5	#include "BedrockConflictMetrics.h"
6	6	#include "BedrockCore.h"
	7	+#include <iomanip>
7	8	
8	9	set<string>BedrockServer::_blacklistedParallelCommands;

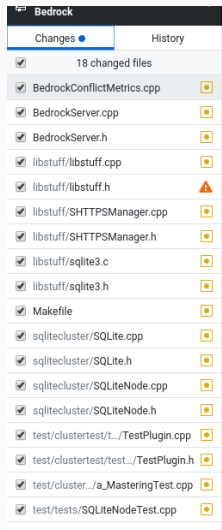
Merge non-automatique : quand il y a des conflits

Message d'erreur :



Merge non-automatique : quand il y a des conflits

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



Merge non-automatique : quand il y a des conflits

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

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()); }
```

Merge non-automatique : quand il y a des conflits

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

 Merge branch 'master' of github.com

description

⌨

Commit to master

ChangesHistory

Merge branch 'master' of github.com:Expensify:Bedrock

Pierre Ortelat committed 16s18 changed files

creating conflict

Merge pull request #364 from Expensify:master

Revert 'Fix repeated journal initializations'

Merge pull request #362 from Expensify:master

Merge pull request #361 from Expensify:master

Cleanup 2

Cleanup 1

Fix port conflict with VM bedrock and ...

Merge branch 'master' into taylor-sqlite...

Working whole fix

Working half-fix

Merge pull request #359 from Expensify:master

BedrockConflictMetrics.cpp

BedrockServer.cpp

BedrockServer.h

Mokole

libstuff/HTTPSManager.cpp

libstuff/HTTPSManager.h

libstuff/libstuff.cpp

libstuff/libstuff.h

libstuff/sqlite3.c

libstuff/sqlite3.h

sqlitecluster/SQLite.cpp

sqlitecluster/SQLite.h

sqlitecluster/SQLiteNode.cpp

sqlitecluster/SQLiteNode.h

test/clusterTest/itestp_.../TestPlugin.cpp

test/clusterTest/itestp.../TestPlugin.h

test/clusterTest/_a_MasteringTest.cpp

test/clusterTest/SQLiteNodeTest.cpp

On peut trouver de l'aide :

Github help :

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

Github desktop help :

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

Installation et configuration

Installer GitHub desktop

- Sur les ordinateurs 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>

Git a besoin de deux informations de base sur vous pour pouvoir travailler efficacement :

- Nom et Prénom
- Email

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...".
- Mettez en collaboration les autres personnes du groupe sur le dépôt. Les autres reçoivent une invitation par mail, ils doivent l'accepter.¹

Pour tout le monde :

- Installez GitHub Desktop, voir liens slide 26.
- Ouvrez GitHub Desktop.
- Clonez le dépôt sur votre ordinateur.

Exercice 1 : partie 2

Une personne :

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

Ensuite, les autres, chacun à son tour :

- Fait un pull
- Regarde l'historique pour vérifier que les changements

Exercice 2 : partie seul

Partie à faire seul (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.
- 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 (cliquer sur "push" si "pull" n'est pas affiché, et cliquez "close" sur le message d'erreur qui s'affiche).
- Resolvez le conflit de merge ensemble pour avoir les deux blagues (il faut éditer le fichier).
- Une fois le merge terminé faire 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 (cliquer sur "push" si "pull" n'est pas affiché, et cliquez "close" sur le message d'erreur qui s'affiche).
- Resolvez le conflit de merge ensemble pour avoir les deux blagues (il faut éditer le fichier).
- Une fois le merge terminé faire un commit et un push.
- La dernière personne fait le pull sur son ordinateur.
- Resolvez 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 :

- Créer un fichier avec Word (.docx) dans le dépôt "*blagues*" précedemment 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.
- Faire push sur le dépôt GitHub en ligne.

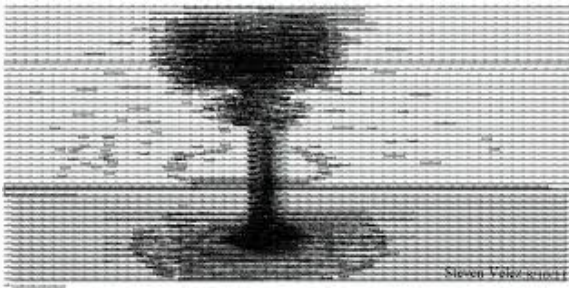
Exercice 3 : partie seul

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!



Exercise 1 : solutions

The screenshot shows the GitHub homepage for a user named Alexandre. The browser address bar shows the URL <https://github.com>. The main heading reads "Learn Git and GitHub without any code!" followed by a subtext: "Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request." Below this are two buttons: "Read the guide" (green) and "Start a project" (grey).

On the left sidebar, the user's profile is shown as "AlexandreFleuret001". Below the profile, there's a section titled "You've been added to the louvainlinux organization!" with some tips for new members. Further down, it shows "louvainlinux forked louvainlinux/Adventure from AlexandreFleuret001/Adventure" and "charlierg forked RailsApp/rails-composer".

On the right side, there's a section titled "Repositories you contribute to" listing several repositories. Below that, the "Your repositories" section is visible, featuring a "New repository" button highlighted with a red circle. The button is green with white text. Below the button is a search bar and a list of repositories including "P4", "autofn", "Adventure", "louvainlinux/Adventure", "Tale", "LINO1123-Calculable", and "Routinisme".

Exercise 1 : solutions

Create a new repository

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

Owner: Repository name:

Create repository names that are short and memorable. Need inspiration? How about [symmetrical-computing-reachline](#).

Description (optional):

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

☐ Private
You choose who can see and control 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.

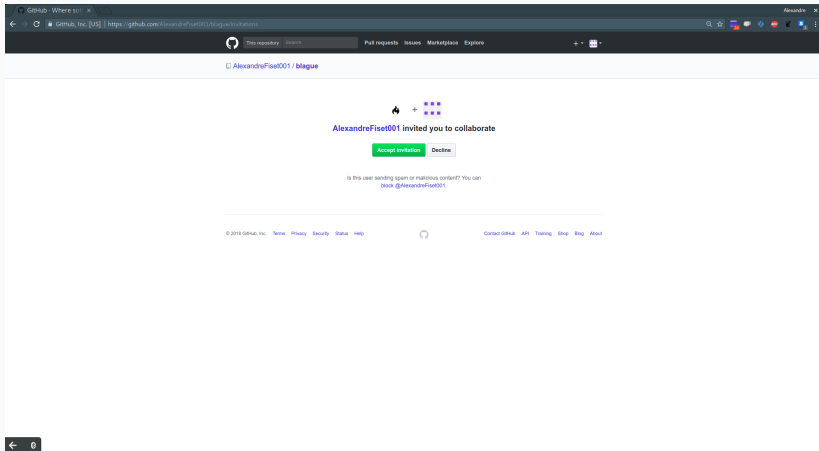
[Create repository](#)

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

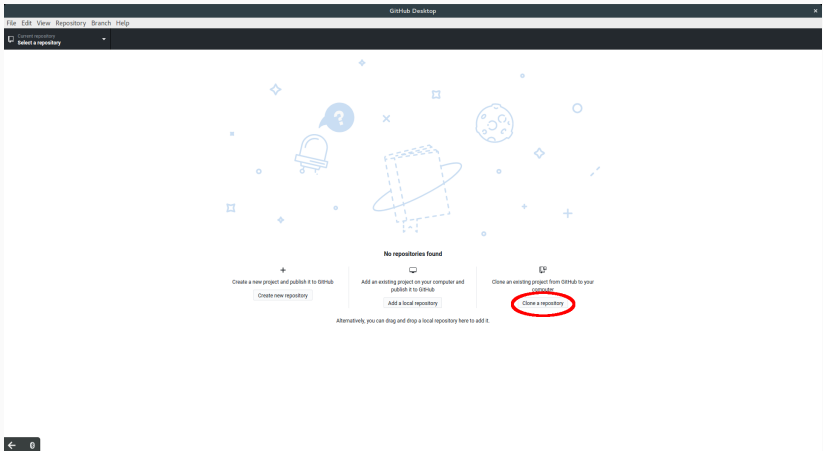
Exercise 1 : solutions

The screenshot shows the GitHub web interface for the 'Collaborators' settings of a repository named 'blague' by user 'AlexandreFise001'. The browser address bar shows the URL: <https://github.com/AlexandreFise001/blague/settings/collaboration>. The page has a top navigation bar with links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below this, there's a sidebar on the left with a list of settings: 'Options', 'Collaborators', 'Webhooks', 'Integrations & services', and 'Deploy keys'. The 'Collaborators' link is highlighted with a red circle and the number '2'. The main content area is titled 'Collaborators' and includes a 'Push access to the repository' section. In this section, there's a 'Copy invite link' button highlighted with a red circle and the number '4', and a 'Cancel invite' button. Below this, there's a search bar with the text 'Search by username, full name or email address' and a note: '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.' The 'Add collaborator' button is highlighted with a red circle and the number '3'. At the bottom of the page, there's a footer with copyright information and links for 'Contact GitHub', 'API', 'Training', 'Shop', and 'Blog About'. A small '0' is visible in the bottom left corner of the screenshot.

Exercise 1 : solutions



Exercise 1 : solutions



Exercise 1 : solutions

AlexandreFiset001 / blague

Code Issues Pull requests Projects Wiki Insights Settings

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

or [HTTPS](https://github.com/AlexandreFiset001/blague.git) [SSH](https://github.com/AlexandreFiset001/blague.git) <https://github.com/AlexandreFiset001/blague.git>

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

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

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

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

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

...or import code from another repository

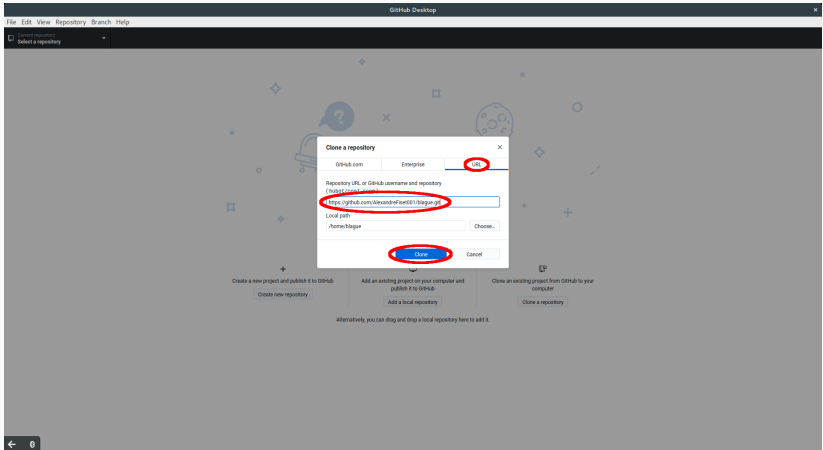
You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

[Import code](#)

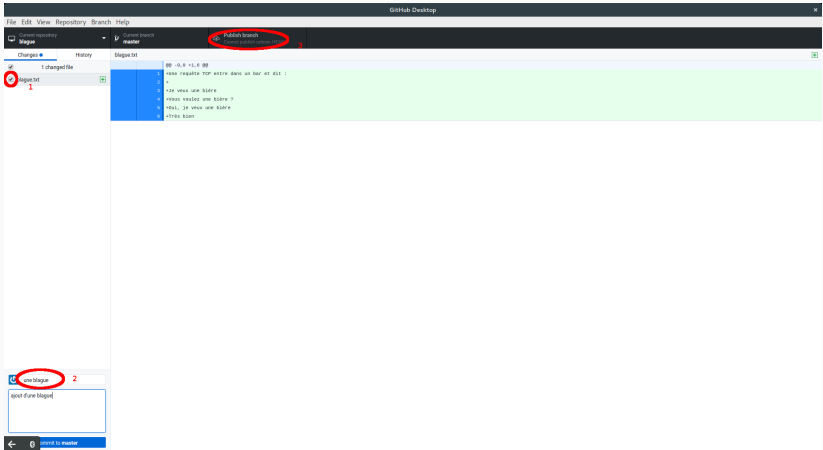
ProTip! Use the URL for this page when adding GitHub as a remote.

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

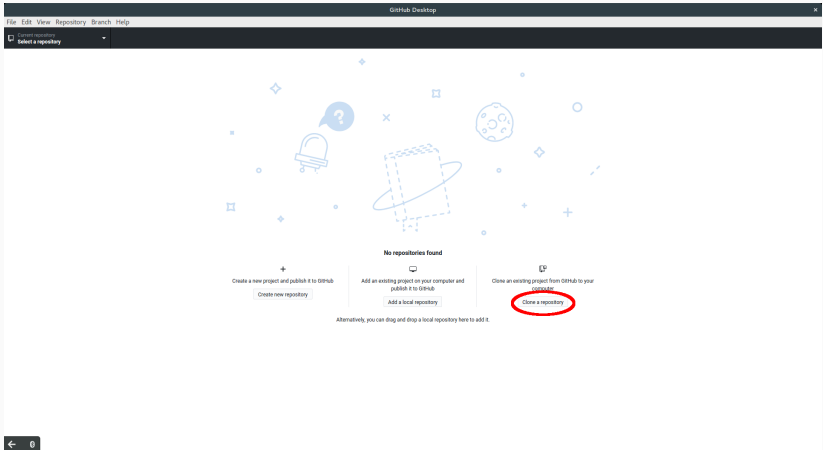
Exercise 1 : solutions



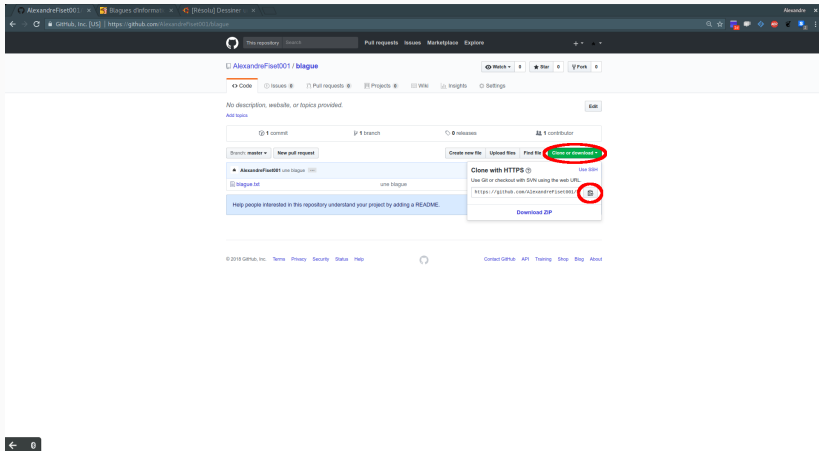
Exercise 1 : solutions



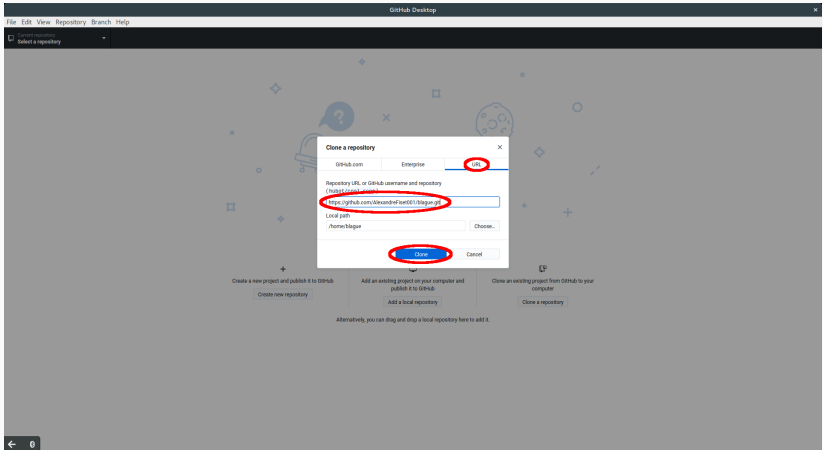
Exercise 2 : solutions



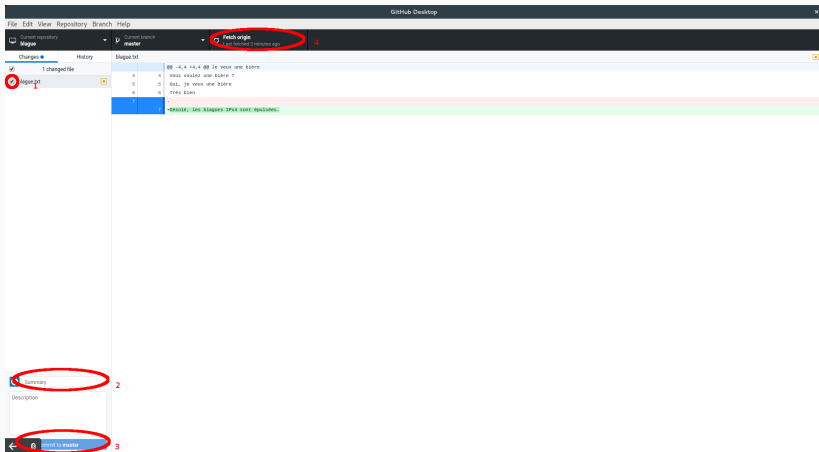
Exercise 2 : solutions



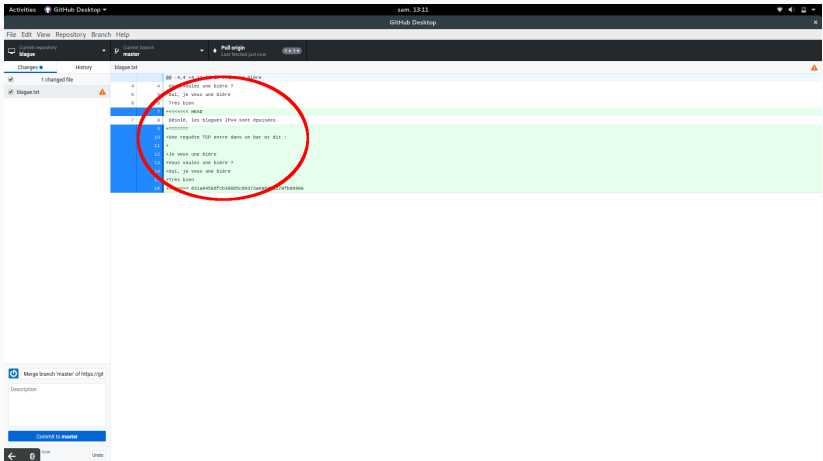
Exercise 2 : solutions



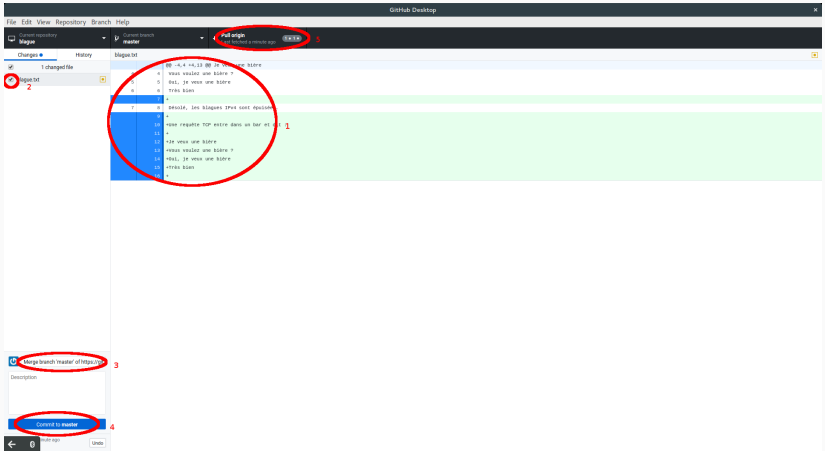
Exercise 2 : solutions



Exercise 2 : solutions



Exercice 2 : solutions



Exercice 2 : solutions

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

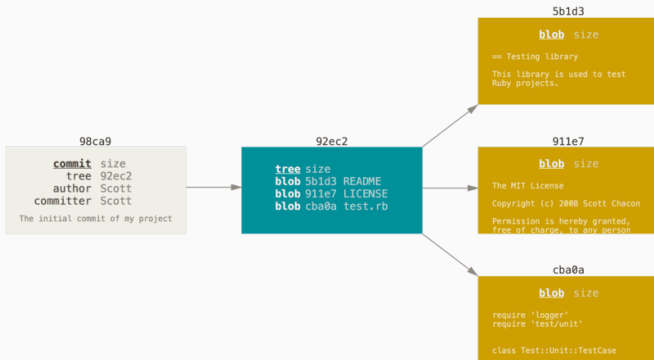
Pour voir tout l'historique, aller sur le site github.com ou bien utiliser une autre interface de git (ex. : commande `git log`).

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 à git (voir plus loin dans les slides).

Fonctionnalités plus avancées

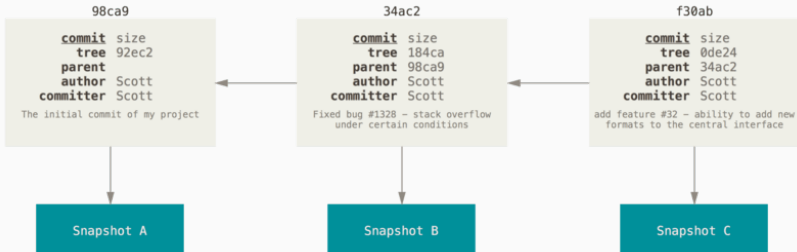
De derrière : les objets git

- 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 overloaded.

 tylerkaraszewski committed 27 days ago ✓



feba803



Mitigate timing issues 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 ▾

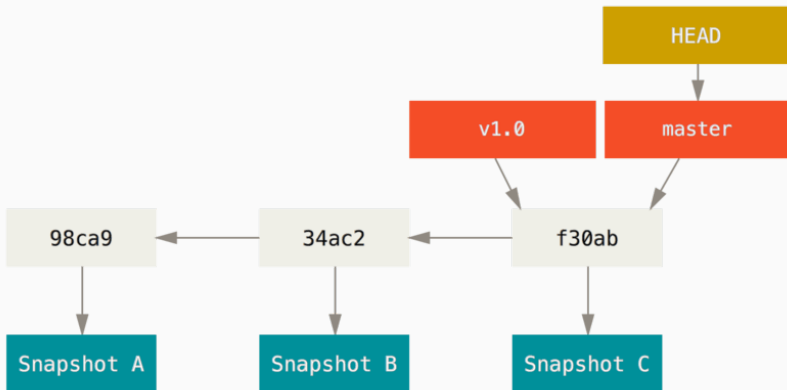
 **tylerkaraszewski** Only log newly queueud commands.

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
📁 sqltecluster	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

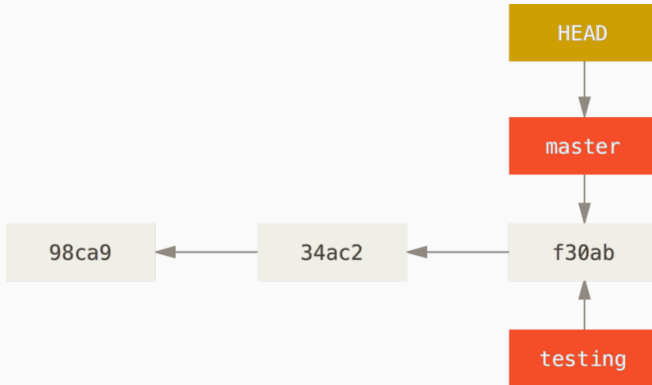
De derrière : les étiquettes

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



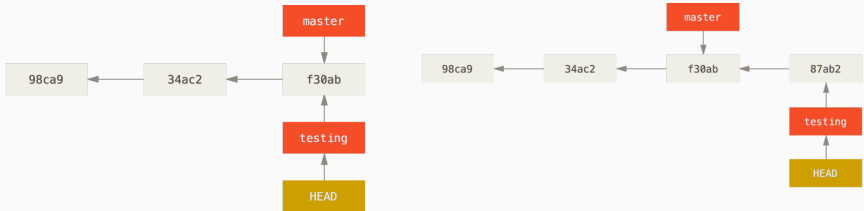
Créer une branche

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



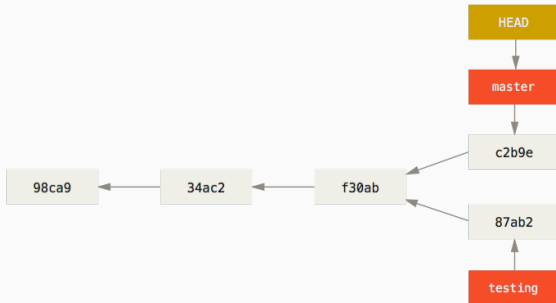
Changer de branche

La branche courante est celle qui suit les nouveaux commits.

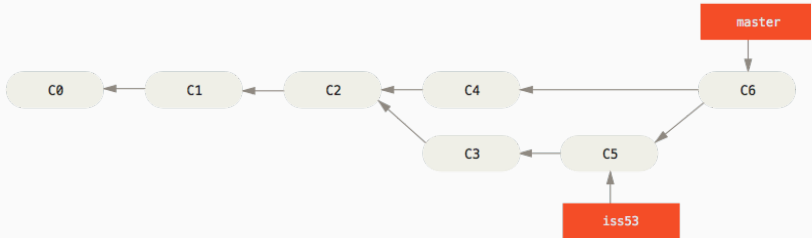
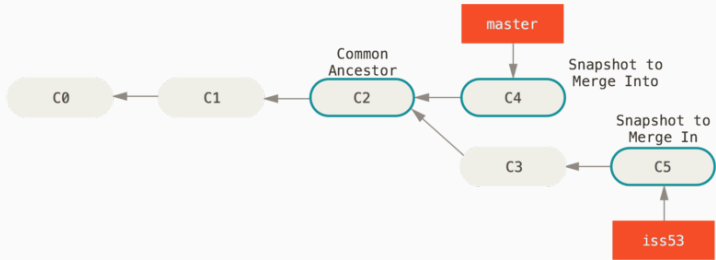


Branches divergentes

- Utilité : travailler sur des modifications indépendantes.



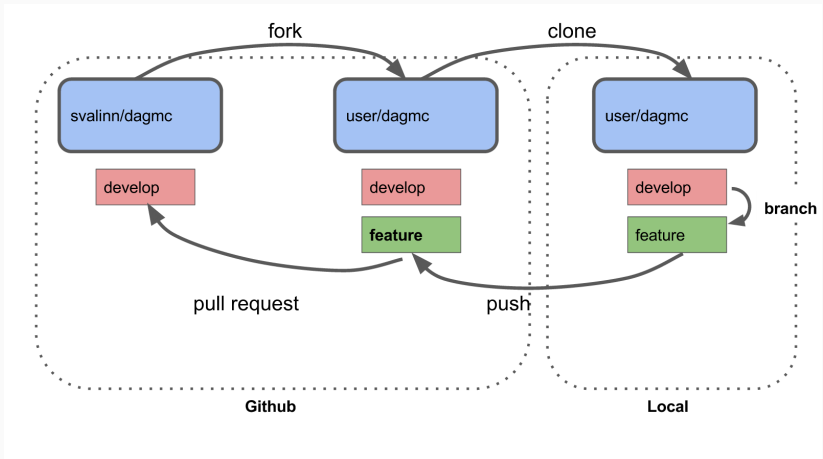
Fusionner des modifications



Parfois il faut résoudre des conflits...

Fork – Pull Request

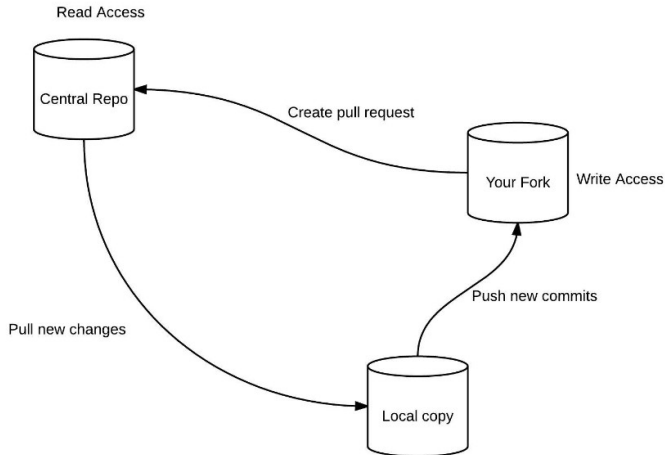
Une autre méthode de collaboration, très utilisée pour des larges projets et/ou projets où la contribution est ouverte à tous.



Fork – Pull Request : Méthode de travail

Voir

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



Forker un dépôt sur GitHub

The screenshot shows the GitHub repository page for `mozilla-mobile / focus-android`. The repository name is at the top left. To the right are buttons for `Watch` (77), `Star` (633), and `Fork` (223). The `Fork` button is highlighted with a green border. Below these are tabs for `Code`, `Issues` (397), `Pull requests` (16), `Projects` (1), `Wiki`, and `Insights`. The repository description is "Firefox Focus: The privacy browser - Browse like no one's watching." Below this are tags: `browser`, `android`, `privacy`, and `mozilla`. At the bottom, a statistics bar shows: 1,384 commits, 9 branches, 34 releases, 43 contributors, and MPL-2.0 license.

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

Créer un dépôt local

Create a new repository

×

Name

Description

Local path

Choose...

☐ Initialize this repository with a README

Git ignore

None ▾

License

None ▾

Create repository

Cancel

Informations et ressources



Pratiquement identiques (tous fonctionnent avec GitHub Desktop).

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