

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 : suivez les instructions à l'adresse <https://louvainlinux.org/git-windows-ucl>.
- Ou installez GitHub Desktop sur votre ordinateur :
 - **Ubuntu** : TODO : Tester RC et lien
 - **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>

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).

- **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,
- ...

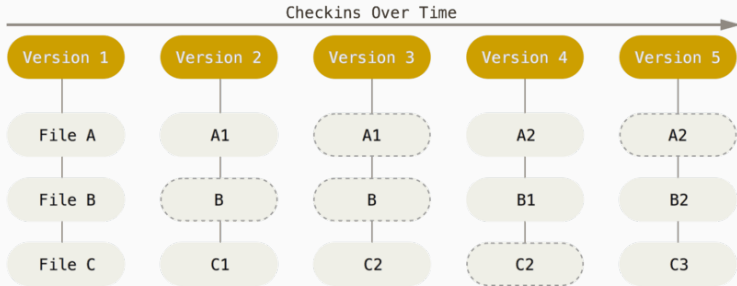
Il y a plein d'"endroits" en ligne où on peut envoyer son travail, GitHub est le plus connu.

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



Repository name

Tux a l'aventure



Great repository names are **Your new repository will be created as 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

Forker un dépôt sur GitHub

The screenshot shows the GitHub repository page for `mozilla-mobile / focus-android`. At the top, there are buttons for `Watch` (77), `Star` (633), and `Fork` (223). 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 the description are tags for `browser`, `android`, `privacy`, and `mozilla`. At the bottom, there is a bar showing statistics: `1,384` commits, `9` branches, `34` releases, `43` contributors, and `MPL-2.0` license. A colorful progress bar is visible at the very bottom.

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

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

 Justenuit ×

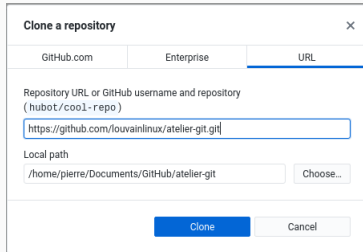
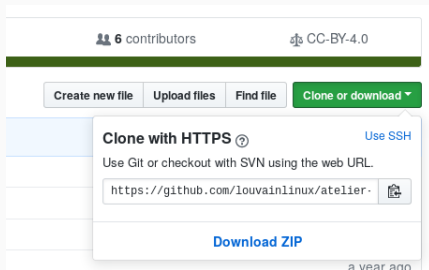
Search by username, full name or email address
You'll only be able to find a GitHub user by their email address if they've chosen to list it publicly. Otherwise, use their username instead.

Add collaborator

Cloner un dépôt sur son PC

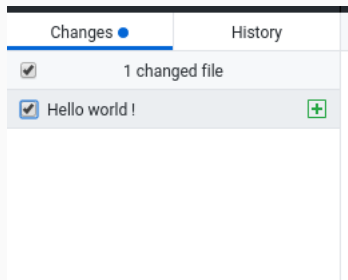
Deux étapes :

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



Pour ouvrir cette fenêtre :
File → Clone repository

Ajouter des fichiers




Voir ce qui est ajouté

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

Remarque : fichier texte vs binaire

- Fichiers texte : programme, \LaTeX ...

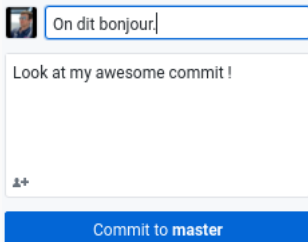
Changes ●	History	Hello world !
<input checked="" type="checkbox"/> 1 changed file		@@ -0,0 +1,2 @@
<input checked="" type="checkbox"/> Hello world ! 		1 +Bonjour tout le monde !
		2 +Comment allez vous ?

- 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 interface for creating a commit. It features a small profile picture of a person with a beard. To the right of the picture 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 #379 from Expensi...

Cole committed 14 hours ago

Don't reply to pseudo-sockets

Tyler Karaszewski committed 15 hours ago

Merge pull request #378 from Expensi...

Cole committed 2 days ago

Missed instance of alarmDuration

Tyler Karaszewski committed 2 days ago

Merge pull request #377 from Expensi...

Cole committed 2 days ago

give up on HTTPS requests if shutdown...

Tyler Karaszewski committed 2 days ago

Only include each crash command onc...

Tyler Karaszewski committed 2 days ago

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 ago

Adds logging suggested by sqllite

Tyler Karaszewski committed 6 days ago

Merge pull request #374 from Expensi...

Cole committed 7 days ago

Continue as soon as we see a comm...

Tyler Karaszewski committed 7 days ago

Add logging to determine if slow time ...

Tyler Karaszewski committed 7 days ago

Merge pull request #373 from Expensi...

Tyler Karaszewski committed 7 days ago

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	00 -265,7 +265,7 00 struct SStopwatch {
266	266	// Accessors
267	267	uint64_t elapsed() { return STimeNow() - startTime.load(); }
268	268	- uint64_t ringing() { return alarmDuration && {elapsed() > alarmDuration.load();} ; }
268	268	+ uint64_t ringing() { return alarmDuration.load() && {elapsed() > alarmDuration.load();} ; }
269	269	
270	270	// Mutators
271	271	void start() { startTime.store(STimeNow()); }

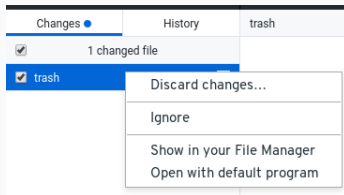
Récupérer un fichier d'un commit passé

TODO : screenshots - on va êtres forcés de faire ça sur Github...

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

↑ **Push origin**
Last fetched 6 minutes ago

File Edit View Repository Branch Help

Current repository **Bedrock** Current branch **master** **Push origin** Last fetched 3 minutes ago

Changes History

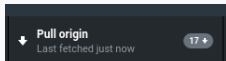
Don't reply to pseudo-sockets

Tyler Karaszewski committed 3cb967f 1 changed file

BedrockServer.cpp

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

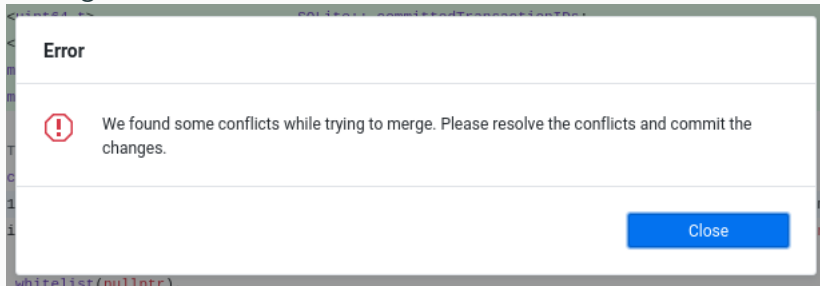
...e-limit
Add logging, change timeout to 10ms - reviewed, merging

BedrockServer.cpp

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

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/clustertest/t.../TestPlugin.cpp	●


Merge non-automatique : quand il y a des conflits

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

	267	+<<<<<<< HEAD
267	268	uint64_t elapsed() { return STimeNow() - startTime; }
268	269	uint64_t ringing() { return alarmDuration.unload() && (elapsed() > alarmDuration); }
	270	+=====
	271	+ uint64_t elapsed() { return STimeNow() - startTime.load(); }
	272	+ uint64_t ringing() { return alarmDuration.load() && (elapsed() > alarmDuration.load()); }
	273	+>>>>>>> ec06209f9747eeca826564f4c1cb566d54bf750
269	274	

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

Changes

History

Merge branch 'master' of github.com:Expensify/Bedrock
Pierre Ottegit committed just now

creating conflict
Pierre Ottegit committed 2 minutes a...

Merge pull request #364 from Expensi...
Tyler Karaszewski committed Jan 31, ...

Revert 'Fix repeated journal initializati...
Cole committed Jan 31, 2018

Merge pull request #362 from Expensi...
Cole committed Jan 31, 2018

Merge pull request #361 from Expensi...
Cole committed Jan 30, 2018

Cleanup 2
Tyler Karaszewski committed Jan 30, ...

Cleanup 1
Tyler Karaszewski committed Jan 30, ...

Fix port conflict with VM bedrock and ...
Tyler Karaszewski committed Jan 30, ...

Merge branch 'master' into tyler-sqlite...
Tyler Karaszewski committed Jan 30, ...

Working whole fix
Tyler Karaszewski committed Jan 30, ...

Working half-fix
Tyler Karaszewski committed Jan 30, ...

Merge branch 'master' of github.com:Expensify/Bedrock
Pierre Ottegit committed 1e92595 16 changed files

BedrockConflictMetrics.cpp	03	03	@@ -83,6 +83,11 @@ bool BedrockConflictMetrics::multiWriteOK(const string& commandName) {
BedrockServer.cpp	04	04	@@ And now that we know whether or not we can multi-write this, see if that's different than the last time we
BedrockServer.h	05	05	@@ checked for this command, so we can do extra logging if so.
Makefile	06	06	@@ if (result != metric_LastCheckOK) {
libstuff/SHHTTPManager.cpp	07	07	@@ // give a fresh start on making this OK again, so that we don't fall back into a DENIED state on the next
libstuff/SHHTTPManager.h	08	08	@@ // check.
libstuff/libstuff.cpp	09	09	@@ metric_results.reset();
libstuff/libstuff.h	10	10	@@ }
libstuff/sqlite3.c	11	11	@@ snprintf("Multi-write changing to " << resultString << " for command '" << commandName
libstuff/sqlite3.h	12	12	@@ << " recent conflicts: " << conflicts << "/" << min(uint64_t)COMMAND_COUNT, totalAttempts)
sqlitecluster/SQLite.cpp	13	13	@@ << " total conflicts: " << metric_totalConflictCount << "/" << totalAttempts << " ");
sqlitecluster/SQLite.h	14	14	
test/clusterTest/testp.../TestPlugin.cpp	15	15	
test/clusterTest/testp.../TestPlugin.h	16	16	
test/clusterTest/_a_MasteringTest.cpp	17	17	
test/Tests/SQLiteNodeTest.cpp	18	18	

On peut trouver de l'aide : TODO : A garder à cet endroit ?

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

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

Installation et configuration

Installer GitHub desktop

- Sur les ordinateur Windows UCL : suivez les instructions à l'adresse <https://louvainlinux.org/git-windows-ucl>.
- 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**
- **E-m-a-i-l**

TODO : screenshots

Linux Windows Mac TODO : screenshots

Pas besoin avec git desktop

Exercices

Exercise 1

TODO

Exercise 2

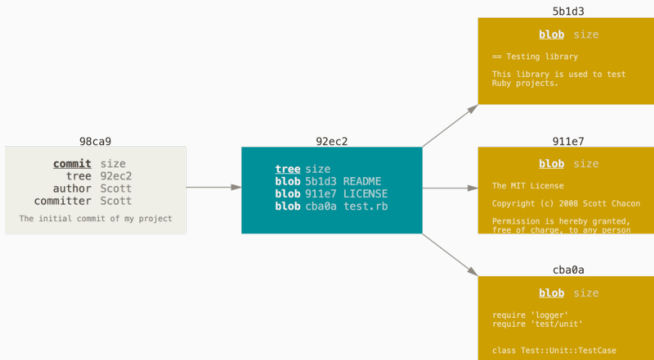
TODO

Fonctionnalités plus avancées

De derrière : les objets git

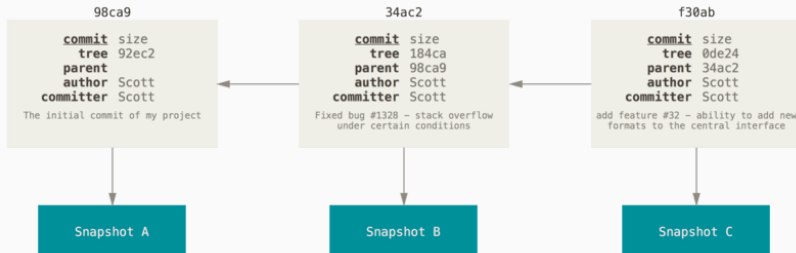
- Chaque commit a un identifiant :

12f87b95caff8cbef5ce0717528d77e27db5669c.



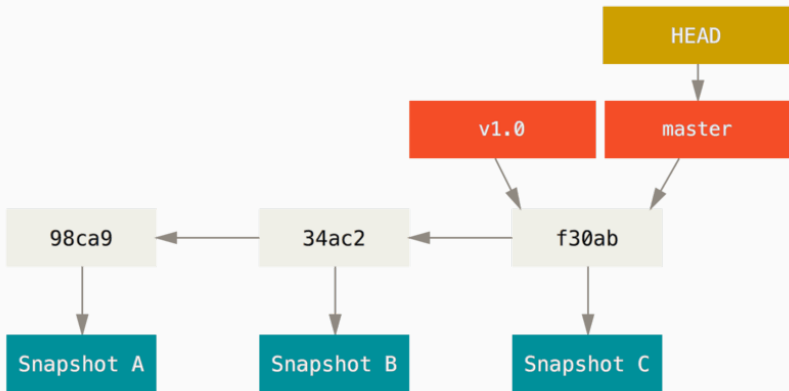
De derrière : les parents

- Chaque commit a un parent.



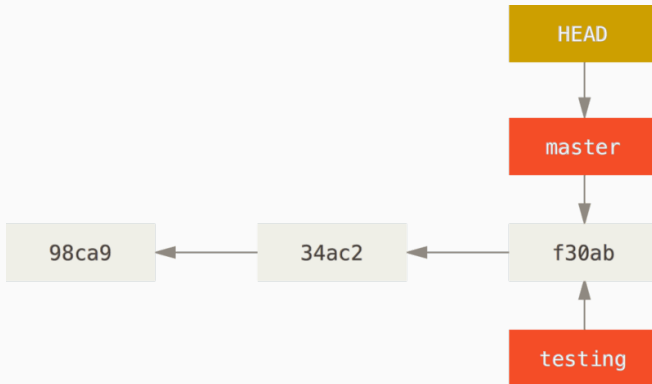
De derrière : les étiquettes

- 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`.



TODO : screenshots

Changer de branche

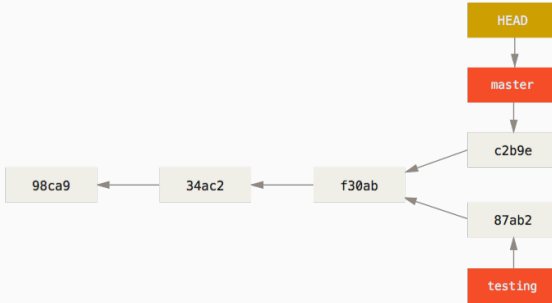
La branche courante est celle qui suit les nouveaux commits.



TODO : screenshots

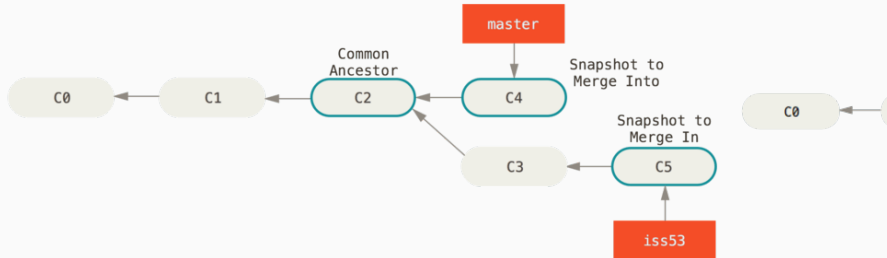
Branches divergentes

- **Utilité** : travailler sur des modifications indépendantes.



TODO : screenshots

Fusionner des modifications

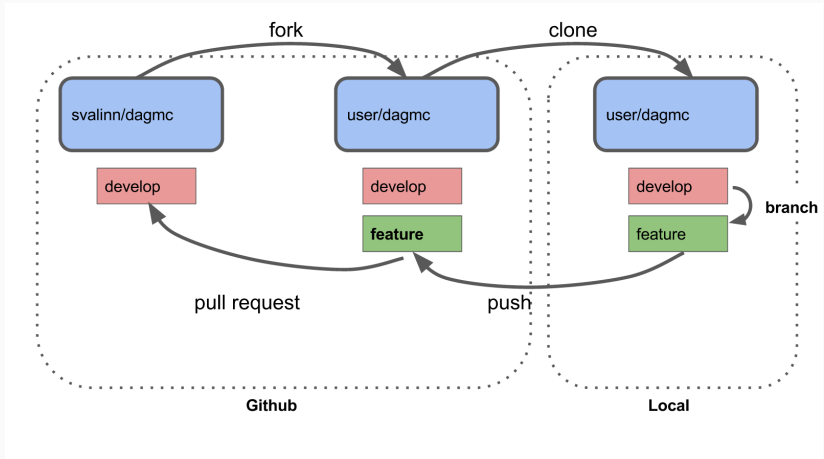


TODO : screenshots

Et parfois il faut résoudre des conflits. . .

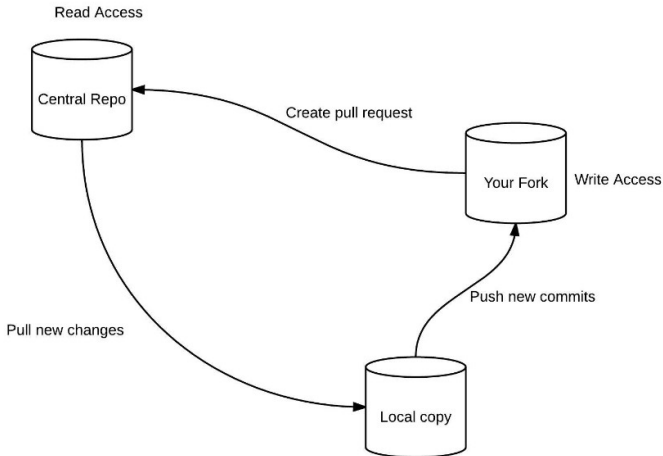
Fork – Pull Request

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

Ajouter un "upstream". TODO : ref vers une explication / voir comment ça marche avec github desktop



TODO : screenshots

TODO : screenshots

Informations et ressources



Pratiquement identiques (tous fonctionnent avec GitHub Desktop).

Dépôts privés gratuits (tout comme sur Gitlab & Bitbucket), et d'autres avantages : <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`
- TODO : autre bon tuto ?

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

Questions ?