Suivre et partager ses sources avec GitLab

Matthieu Boileau, Alexis Palaticky

CNRS - Université de Strasbourg

8 novembre 2016

&

GitLab

Outline

Objectif de l'exposé

- **Pour les non initiés :** à quoi servent git et GitLab ?
- **Pour tout le monde** :
  - que peut-on faire avec GitLab ?
  - comment administre-t-on un serveur GitLab ?
- **Répondre aux questions suivantes :**
  - ai-je intérêt à utiliser Git dans mon travail ?
  - mes utilisateurs y ont-ils intérêt ?
  - ai-je intérêt à déployer un serveur GitLab dans mon unité ?

Pourquoi utiliser un suivi de version ?

- **Enregistrer** les modifications d'un jeu de fichiers au cours du temps
- Rester **réversible** :
  - pouvoir retourner à une version antérieure,
  - comparer avec une version antérieure
- **Documenter** les modifications (date, auteur et message d'accompagnement)
- Un logiciel de suivi de version (VCS pour *Version Control System* en anglais) comme Git gère très bien tout projet qui se présente sous la forme de **fichiers sources**
- C'est le meilleur moyen de **collaborer** sur des sources !

Pourquoi utiliser un suivi de version ?

Ce que Git gère très bien :

- ✓ les scripts ou code de calcul
- ✓ les documents LaTeX (cet exposé est sur GitLab !)
- ✓ les fichiers texte de configuration
- ✓ les sources html
- ✓ etc.

Ce que Git gère mal :

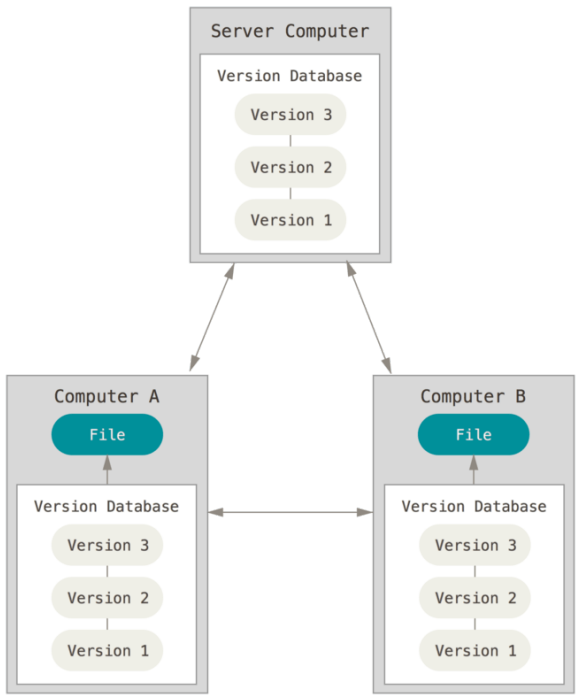
- ✗ les gros fichiers binaires
- ✗ les documents Microsoft Office ou OpenOffice
- ✗ le texte formaté en général
- ✗ les bases de données (type mysql)
- ✗ etc.

Comparaison avec un système de partage de type Owncloud ou Seafile

	Owncloud	Git
Type de fichiers	✓ tous types	✓ suivi pour les fichiers sources ✗ pas de suivi pour les binaires ✗ pas adapté aux gros fichiers (sauf avec git-lfs)
Suivi de version	✗ très limité	✓ outil avancé
Partage	✗ modèle centralisé uniquement ✗ synchronisations essentiellement automatiques	✓ modèle distribué ✓ on contrôle les synchronisations
Prise en main	✓ très simple	✗ demande un apprentissage

Un suivi de version distribué

- les clients possèdent un miroir complet de la base de données du serveur
- on peut travailler en mode déconnecté et synchroniser quand on le souhaite
- indirectement, on crée des sauvegardes multiples

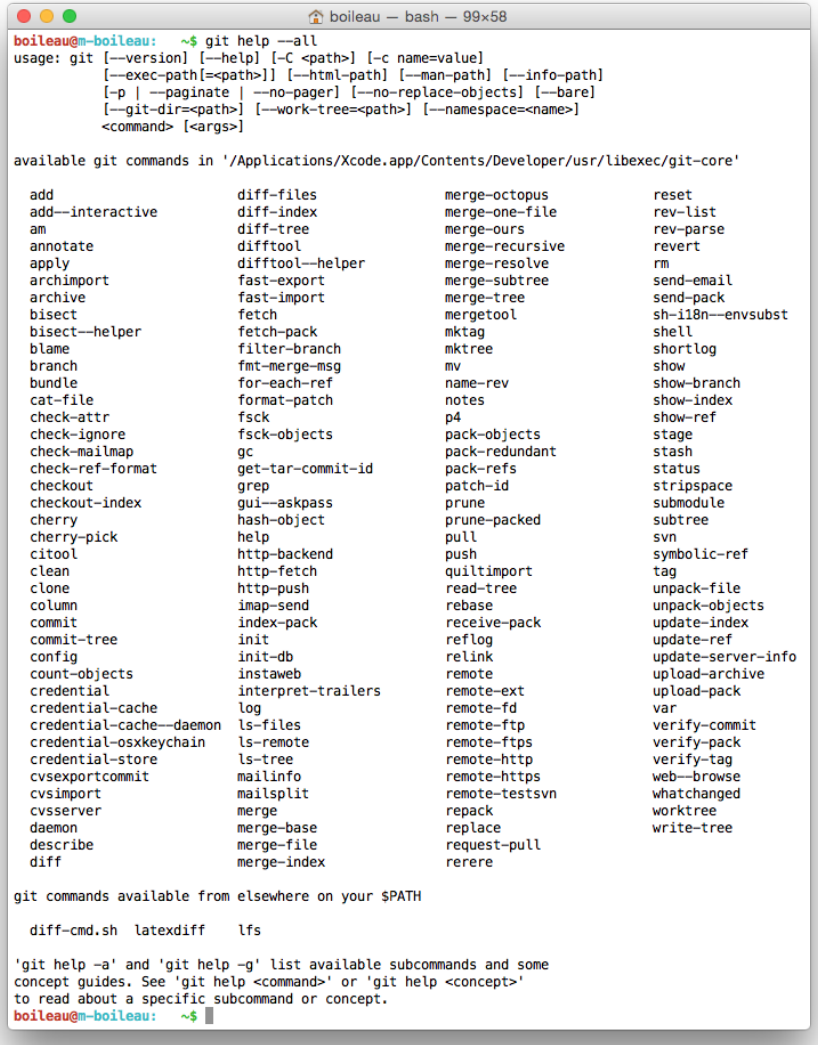


(Source: Pro Git book `http://git-scm.com/book`)

Git en pratique

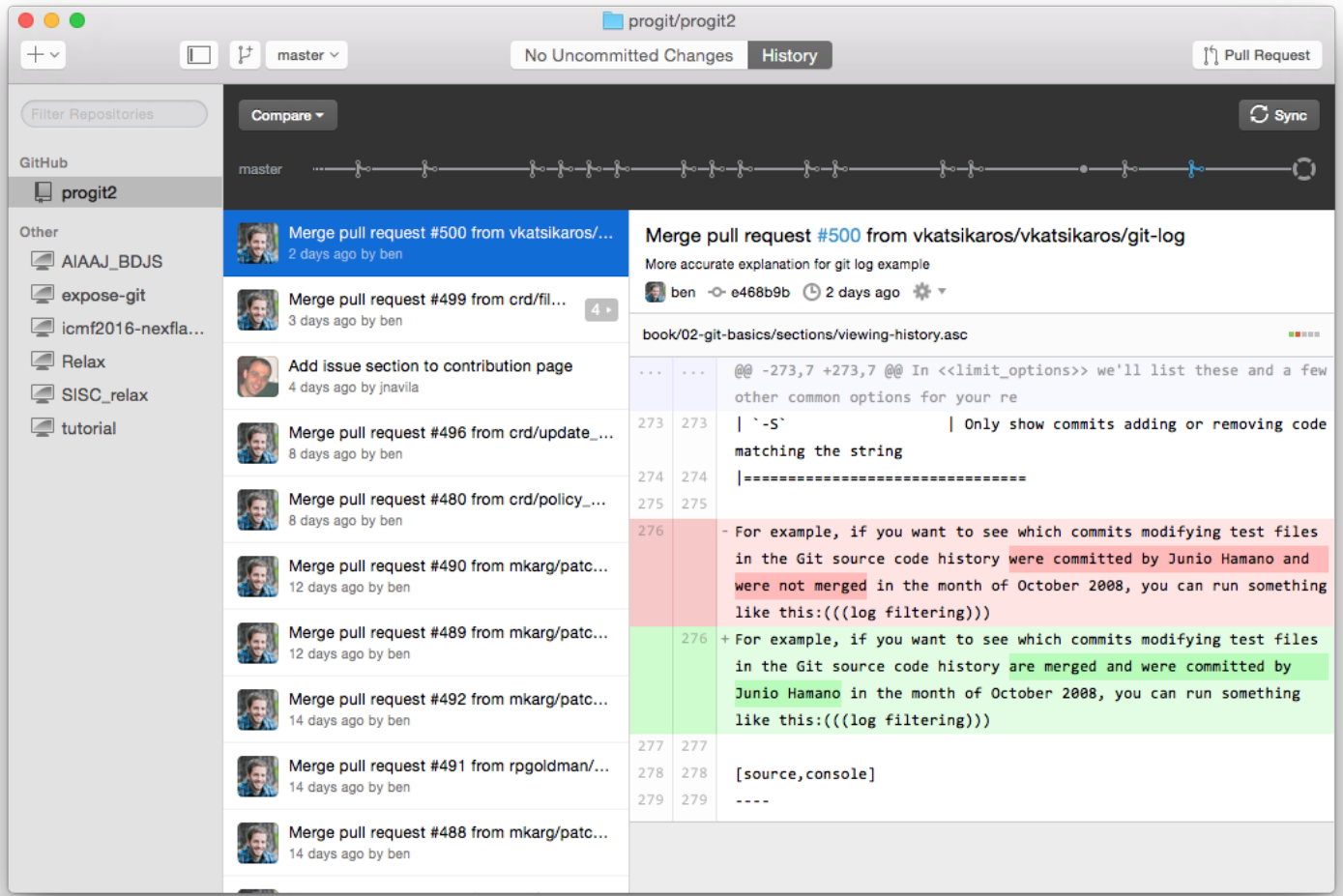
Git en ligne de commande dans le Terminal :

```
git add
git commit -m "My commit message"
git status
git log
git push
git pull
git checkout
git diff
etc.
```



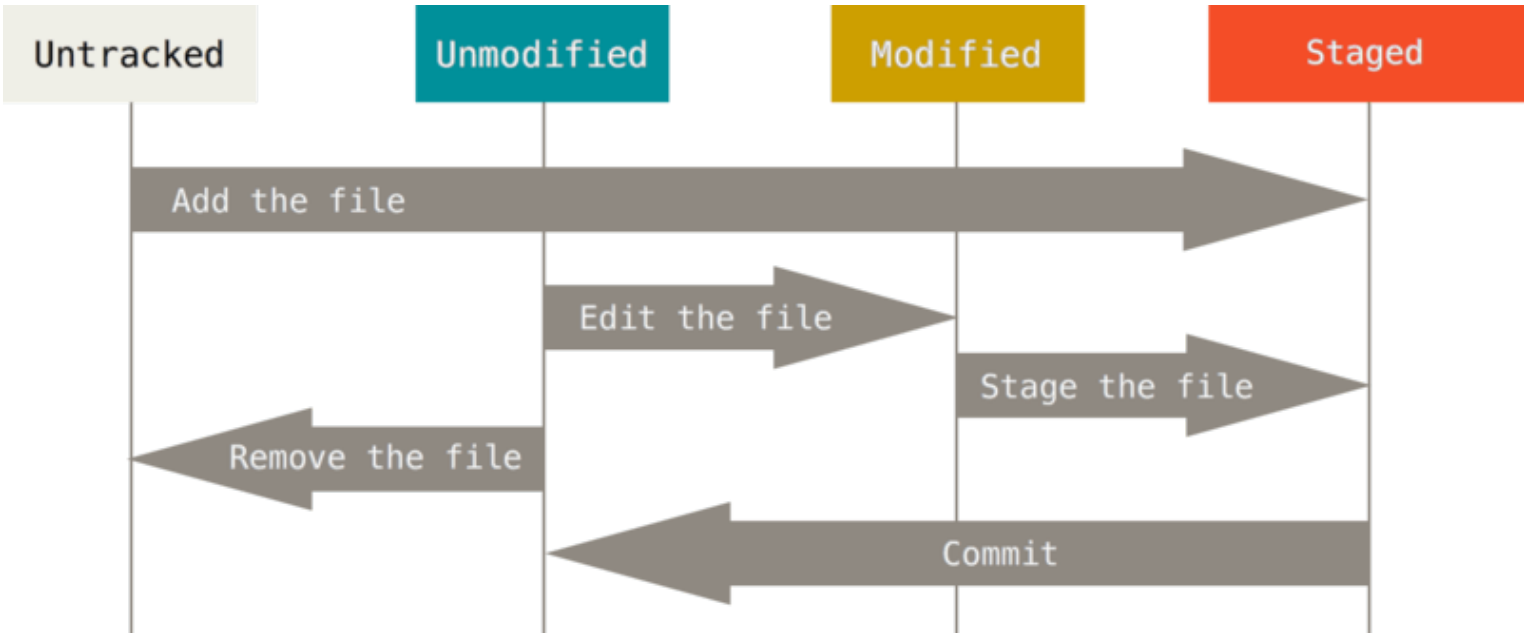
Git en pratique

Avec une interface graphique très simple comme GitHub Desktop, on couvre ≈ 90% de l'utilisation courante de git :



Les quatre statuts des fichiers suivis

- Le cycle de vie d'un fichier suivi avec Git

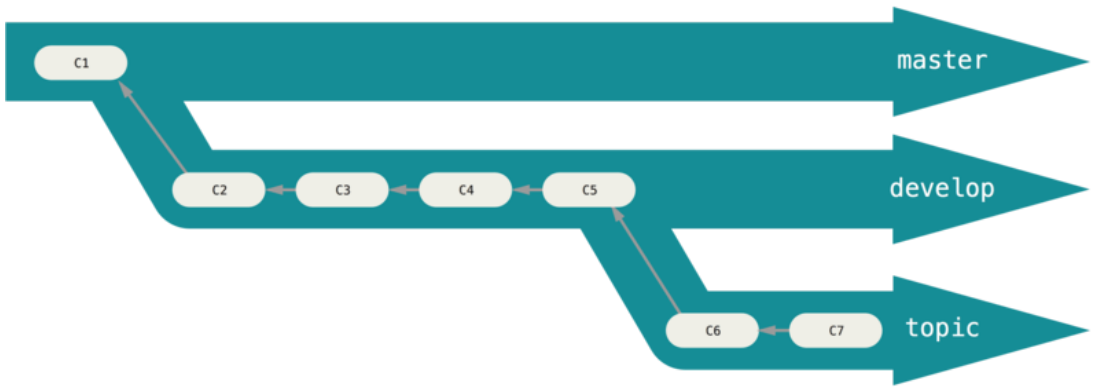


(Source: Pro Git book `http://git-scm.com/book`)

- Les fichiers qui ne sont pas des sources (fichiers objets, fichiers de compilations, exécutables, etc.) peuvent être ignorés.

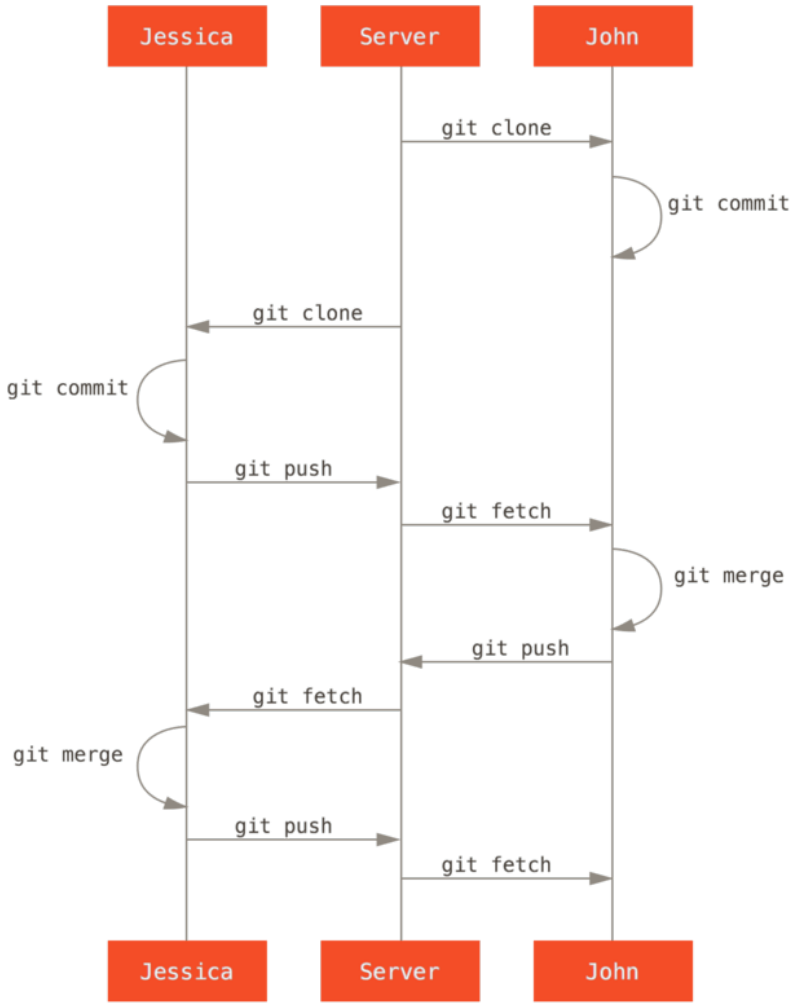
Git et le système des branches

- Git permet de créer et fusionner très facilement des branches
- un système de branches permet de préserver une version stable (branche *master*) sans limiter les développements (branche *develop*)
- les branches sont particulièrement utiles pour le travail collaboratif et par sujet *(topic)*.



(Source: Pro Git book `http://git-scm.com/book`)

Travail collaboratif avec Git : le workflow typique d'une petite équipe



(Source: Pro Git book `http://git-scm.com/book`)

En pratique :

- Côté serveur : **GitLab**
- Côté clients (John et Jessica) : ligne de commande ou client graphique (GitHub Desktop, par exemple)

Travail collaboratif avec Git

- Même si **Git** est basé sur un modèle distribué, le partage se fait essentiellement via un **serveur Git.**

Comparaison des services d'hébergement Git en version gratuite

Github	Bitbucket	GitLab
✓ très gros projet (138+ millions dépôts, 600 employés)	✓ très gros projet	✓ projet pérenne et dynamique (132 employés, 33 pays)
✓ Grande interopérabilité avec d'autres outils	✓ Git & Mercurial	✓ pas de limite dans la version hébergée
✗ dépôts publics uniquement	✓ intégration dans les produits Atlassian	✗ des lenteurs sur le site gitlab.com
✗ pas d'instance privée	✗ 5 utilisateurs max/dépôt	✓ instance privée opensource
	✗ pas d'instance privée	✓ outils d'intégration continue natifs

(novembre 2016, d'après `http://comparegithosting.com`)

Parmi ces 3 géants, **GitLab** est la seule solution qui permet d'héberger une instance privée (problème de l'hébergement des données de recherche sur des clouds privés).

Gitlab comme serveur git

GitLab

Installation et administration

Gitlab comme serveur git

- Outil opensource de gestion de projets git (licence MIT)
- Pourquoi GitLab à l'IRMA ?
  - Besoin de travail collaboratif sur du code ou des publications
  - Pour avoir la maîtrise sur le paramétrage et la création des comptes
  - Gérer des fonctionnalités avancées comme l'intégration continue ou git LFS

Les différentes versions

- GitLab Community Edition (CE) : version libre
- GitLab Enterprise Edition (EE) : version payante
- GitLab Continus Integration (CI) : intégrée dans CE et EE depuis GitLab 8.0
- `https://about.gitlab.com/features/#compare`

Gitlab comme serveur git

GitLab à l'IRMA

- `https://gitlab.math.unistra.fr`
- Serveur sauvegardé en interne (VDP et Netbackup)
- Mise à jour de GitLab et mises à jour systèmes régulières
- Serveur accessible de l'extérieur, possibilité de comptes externes

Gitlab comme serveur git

Installation de GitLab : les recommandations de `doc.gitlab.com`

Les OS supportés

- Ubuntu
- Debian
- CentOS
- Red Hat Enterprise Linux (use the CentOS packages and instructions)
- Scientific Linux (use the CentOS packages and instructions)
- Oracle Linux (use the CentOS packages and instructions)

Gitlab comme serveur git

Installation de GitLab : les recommandations de `doc.gitlab.com`

Les OS non supportés

- OS X
- Fedora
- Gentoo
- FreeBSD
- Windows

Gitlab comme serveur git

Installation de GitLab : les recommandations de `doc.gitlab.com`

La configuration hardware

- 1 core works supports up to 100 users but the application can be a bit slower
- 2 cores is the recommended number of cores and supports up to 500 users
- 2GB RAM is the recommended memory size and supports up to 100 users
- 4GB RAM supports up to 1,000 users

Gitlab comme serveur git

Installation de GitLab à l'IRMA

- Gitlab-ce version 8.13.3
- Machine virtuelle dédiée (Ubuntu Serveur Ubuntu 14.04.4 LTS)
- 2 processeurs
- 8 Go de ram
- 60 Go de stockage

Gitlab comme serveur git

Prérequis

- curl
- openssh-server
- ca-certificates
- postfix

Installation

```
apt-get install curl openssh-server ca-certificates postfix

curl https://packages.gitlab.com/install/repositories/gitlab/
    gitlab-ce/script.deb.sh | sudo bash sudo apt-get install
    gitlab-ce
```

gitlab-ce contient toutes les dépendances nécessaires : Nginx, Ruby, Postgres...

Gitlab comme serveur git

Configuration

- Un fichier central : /etc/gitlab/gitlab.rb.
- C'est dans ce fichier que l'on définit les principaux paramètres :
  - url du serveur
  - certificats électroniques
  - jonction avec un annuaire LDAP...

Gitlab comme serveur git

Exemple de l'installation du certificat Digicert :

- Copie des fichiers du certificat dans : /etc/gitlab/ssl
- Modification du fichier /etc/gitlab/gitlab.rb

```
external_url 'https://gitlab.math.unistra.fr'
nginx['redirect_http_to_https'] = true
nginx['ssl_certificate'] = "/etc/gitlab/ssl/fichier_certificat.pem"
nginx['ssl_certificate_key'] = "/etc/gitlab/ssl/fichier_certificat.key"
```

- Validation de la nouvelle configuration pour prise en compte des modifications : gitlab-ctl reconfigure

Gitlab comme serveur git

Sauvegarde de la base

- Une commande à placer dans une crontab :

  gitlab-rake gitlab:backup:create

- Résultat :

  Dépôt d'un fichier xxxxxx_gitlab_backup.tar dans /var/opt/gitlab/backups

Gitlab comme serveur git

Mises à jour de GitLab

- Elles sont fréquentes (tous les 15 jours)
- Utilisation de la commande `apt-get update`
- Déclenchement automatique d'une sauvegarde de la base avant la mise à jour

GitLab comme serveur git

L'interface d'administration de GitLab

- Connexion sur l'*admin area*
- La gestion des utilisateurs
- L'affichage de messages pour les utilisateurs
- Le menu settings :
  - Nombre de projets par défaut pour les utilisateurs
  - Statut par défaut du projet (private, public)
- Les *runners* d'intégration continue
- Tableau de bord : observer et mesurer l'activité du serveur gitlab
- Pour connaître les nouveautés de GitLab, la newsletter `https://about.gitlab.com/blog/`

Gitlab comme serveur git

GitLab

Exploitation par l'utilisateur

Exploitation par l'utilisateur

Les fonctionnalités de GitLab

- Création rapide de projets avec gestion des droits (Public, privé,...)
- Groupes de projets
- Historique des commits
- Edition en ligne
- Outils annexes :
  - statistiques
  - wiki
  - gestionnaire de tickets
  - notifications par mail
- Intégration continue avec `gitlab-ci`

Démonstration en ligne sur `https://gitlab.math.unistra.fr`

Installer son serveur GitLab ?

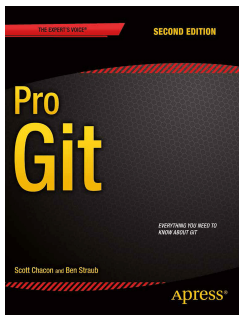
Pour accéder à un serveur GitLab, des solutions institutionnelles existent comme `https://gitlab.unistra.fr` alors...

pourquoi déployer un serveur GitLab local ?

- pour maîtriser l'accès au service (en particulier pour les collaborateurs extérieurs)
- pour maîtriser la maintenance (mise à jour, ajout de fonctionnalités, etc.)
- besoin faible en ressources informatiques
- effort modéré pour l'administration
- politiquement : on identifie le contenu à l'entité qui l'héberge

Documentation et bonnes pratiques

Documentation



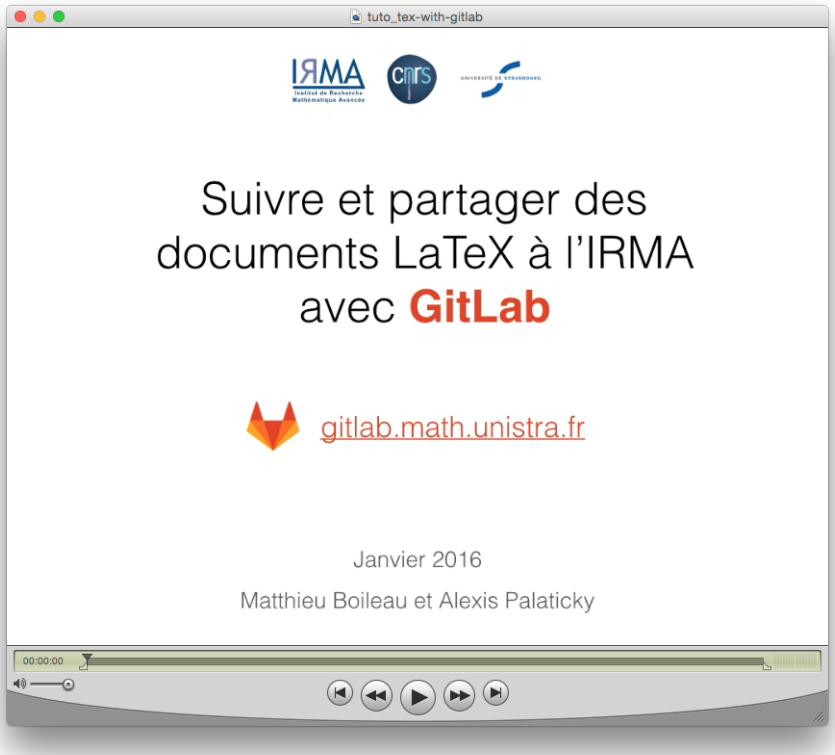
ProGit book : une référence (libre)

- Pro Git book (version française) : `https://git-scm.com/book/fr`
- la documentation officielle de Git : `http://git-scm.com/documentation`
- un manuel concis : `http://gitref.org/index.html`
- la doc GitLab et son aide contextuelle : `https://gitlab.math.unistra.fr`

Quelques recommandations

- En LaTeX, ne pas écrire plus d'**une phrase par ligne**
- Faire de **nombreux commits** (Git est fait pour ça) contenant des modifications **petites et cohérentes** plutôt que l'inverse
- Utiliser les branches !

Documentation et bonnes pratiques



Pour ceux qui sont intéressés :
un tutoriel vidéo de 15 min pour suivre des sources LaTeX avec GitLab.