

# 1 Introduction

## 1.1 Histoire

### 1.1.1 Versionnage de code ?

Avez vous déjà travaillé sur un code en groupe ? Avez vous déjà tenté de programmer un logiciel sur plusieurs versions ? Si oui, vous vous êtes sans doute rendu compte qu'il était nécessaire d'avoir un outil pour pouvoir facilement travailler sur le même code au même moment. Ou alors pouvoir avoir un historique des modifications en fonction des versions. Vous avez donc besoin d'un outil de versionnage de code.

### 1.1.2 Outils de versionnage

Il existe de nombreux outils de versionnage de code. Les 3 plus connus sont sans doute Git, Mercurial et SVN.

**SVN (subversion)** Ce fut le plus utilisé pendant longtemps. Développé par la fondation Apache, il s'agit d'une amélioration d'un programme nommé CVS (très peu utilisé aujourd'hui). Le plus gros problème de SVN est qu'il s'agit d'un système centralisé. Un serveur contient donc le code, des clients travaillent dessus. Il n'y a qu'un seul versionning.

**Mercurial** Contrairement à SVN, il s'agit d'un système décentralisé. Chacun possède son propre repository et publie son code sur le repository public. Une autre différence avec SVN est qu'il utilise la notion de changeset. C'est à dire qu'il préfère garder en mémoire les changements appliqués que les versions des fichiers.

**Git** Git possède très peu de différences avec Mercurial, mais l'histoire a fait qu'il c'est plus imposé que mercurial (qui est quand meme utilisé dans pas mal de projets dont ceux de Mozilla, même si git est possible). C'est en partie grâce à Linus Torvalds qui en a fait la pub et des projets comme github que nous allons utiliser dans le reste du tutorat. Pour plus d'informations sur les différences vous pouvez vous référer à ces articles : <http://importantshock.wordpress.com/2008/08/07/git-vs-mercurial/>, <http://www.rockstarprogrammer.org/post/2008/apr/06/differences-between-mercurial-and-git/>.

## 1.2 Installation

### 1.2.1 Windows

Lancez le programme que vous pouvez trouver sur cette page : <http://msys-git.github.io>

### 1.2.2 Linux

`apt-get install git` ou `yum install git` selon la distribution

### 1.2.3 Mac OS

```
sudo port install git-core +svn +doc +bash_completion +gitweb
```

## 1.3 Github/Gitlab

Pour la suite de ce tutorat, il vous faut créer un compte github : <https://github.com/>. Vous pouvez aussi vous intéresser à gitlab : <https://about.gitlab.com/>

## 2 Manipulation basique de git

### 2.1 Premier commit

#### 2.1.1 créer le dépôt

#### 2.1.2 pull

#### 2.1.3 readme

#### 2.1.4 add/diff/ignore/commit

#### 2.1.5 push

### 2.2 Historique

## 3 Manipulation avancée

### 3.1 Méthode de développement

### 3.2 branch

#### 3.2.1 créer sa branche

#### 3.2.2 fusionner

#### 3.2.3 résoudre un conflit

#### 3.2.4 savoir qui a fait quoi

### 3.3 Se positionner sur un commit

### 3.4 Contribuer à un autre repository