Git

- Syncro Git

Syncro Git

Créer et cloner un répertoire Git

Le répertoire Git est le répertoire central de tout projet géré et constitue la zone commune à tous les participants permettant de régler l'intégralité du contrôle des versions. Votre première étape dans Git consistera donc à **créer un tel répertoire principalou à le cloner** (sous la forme d'une copie de travail), dans la mesure où vous vous connectez à un projet dont le développement commun est déjà géré à l'aide de Git.

Si vous souhaitez reconfigurer le contrôle de versions ou si vous venez tout juste d'installer cet outil pour apprendre à utiliser Git, vous devrez tout d'abord créer un répertoire. Pour ce faire, allez dans le **répertoire local désiré** sur votre appareil à l'aide de « cd » (***c**hange **d**irectory*) :

```
git init
```

Si le répertoire Git existe déjà pour votre projet, vous aurez simplement besoin de l'**adresse Web ou réseau de ce répertoire** pour créer une **copie de travail** sur votre ordinateur avec la commande « **git clone** » :

Vérifier le statut du répertoire et ajouter de nouveaux fichiers à la gestion de versions

Une bonne organisation du répertoire de travail fait partie des bases essentielles de Git. Ce répertoire vous permet non seulement de proposer vos propres **modifications et de nouveaux ajouts** à un projet qui sont alors repris via commit (« *partage* »), mais aussi d'acquérir des **informations sur les activités d'autres utilisateurs**. Vous pouvez vérifier l'actualité de votre copie de travail à l'aide de la commande suivante :

```
git status
```

En cas de création d'un nouveau répertoire ou de correspondance absolue du répertoire principale et de la copie de travail, vous êtes généralement informé du fait que le projet ne comporte aucune modification (« *No commits yet* »). Git vous indique par ailleurs que vous n'avez actuellement partagé aucune modification pour le prochain commit (« *nothing to commit* »).
Afin d'ajouter un nouveau **fichier à la gestion des versions** ou de signaler un fichier édité **pour le prochain commit**, utilisez la commande « **git add** » sur ce fichier (il doit se trouver dans le répertoire de travail). Dans notre tutoriel Git, nous avons ajouté à titre d'exemple un document texte intitulé « Test » :

Valider les modifications via commit et les reprendre dans le HEAD

L’ensemble des modifications que vous avez enregistrées pour la gestion des versions (de la façon décrite à la section précédente) doivent toujours être validées par un commit pour être **reprises dans le HEAD**. Le HEAD est un type d’index qui renvoie au dernier commit ayant pris effet dans l’environnement de travail Git actuel (également appelé « branche »). Pour cette étape, la commande est la suivante :

```
git config --global user.email "garfieldtux@gmail.com"
git config --global user.name "garfieldtux"
git config --global core.editor "nano"
git push origin master
git commit -a
```

Reprendre des commits dans le répertoire principal

Jusqu’à présent, nous vous avons montré dans ce tutoriel Git comment enregistrer les modifications sous forme de commit dans le HEAD du répertoire local. Toutefois, pour qu’elles soient reprises **dans le répertoire principal**, il est nécessaire de saisir la commande suivante :

```
git push origin master
```

Avec ce code, **Git transmet automatiquement tous les commits créés**, qui jusqu’à présent se trouvaient uniquement dans la copie de travail, **dans le répertoire principal** également appelé « master ». Si vous remplacez ce nom dans le code indiqué par une autre branche (branche de projet), les fichiers sont alors directement envoyés à cet endroit.