

# TP Git

Thomas Clavier

## 1 configuration

Configurez votre nom (user.name) votre email (user.email) et le serveur mandataire (http.proxy) dans git.

## 2 L'espace de travail

À partir de là nous allons construire un blog de promotion.

### 2.1 Initialisation

Créez un répertoire de travail et initialisez le comme un dépôt git local.

### 2.2 Historique de version

Dans ce blog, tous les articles sont rédigé en utilisant le format Markdown (<http://daringfireball.net/projects/markdown/syntax>).

#### 2.2.1 Premier changement

Dans l'espace de travail que vous avez initialisez comme dépôt local, créez un sous répertoire «content».

Avec votre voisin choisissez un sujet, puis créez le même article «content/[votre-sujet].md» chacun dans votre dépôt local avec le contenu suivant :

```
+++
date = "2015-03-10T12:00:00+01:00"
draft = true
title = "le sujet"
```

```
+++
```

```
# Le premier titre
```

Vous pouvez par exemple faire un article avec les réponses à ce TP.

L'ajouter dans l'index comme fichier à suivre, puis validez le changement et enfin observez l'historique des changements, quelles sont les informations disponibles ? À quoi correspond chaque champ ?

### 2.2.2 Second changement

Ajoutez quelques informations dans votre article puis validez un nouveau commit.  
Comment obtenir le diff entre les 2 derniers commits ?

### 2.2.3 Suppression

Créez un article «content/satoshi-tajiri.md» puis validez un nouveau commit. Supprimez l'article «content/satoshi-tajiri.md» puis validez un nouveau commit.  
Que voit-on dans l'historique ?

## 3 Partager

### 3.1 Via un dépôt partagé

Indiquons à Git que nous souhaitons échanger des données avec le dépôt distant suivant : <http://git.iut.azae.net/git/tp-git.git> Nous appelons ce dépôt «origin».  
Puis envoyez l'ensemble des modifications de la branche courante (master) vers le dépôt «origin»  
Que se passe-t-il ?  
Visitez <http://git.iut.azae.net/> pour admirer votre travail.  
Expliquez le résultat de la commande suivante :

```
git log --oneline --graph --decorate
```

Si l'on souhaite obtenir une copie de travail d'un dépôt distant existant il est possible d'utiliser la commande :

```
git clone url
```

Quelles sont les formes d'URL possibles ?  
Lançons la génération des pages html :

```
./bin/hugo --theme=hyde --buildDrafts
```

Que donne la commande suivante ? Pourquoi ?

```
git status
```

Est-il normale d'enregistrer du code généré ?  
Pour ignorer tout le répertoire de génération modifions le fichier .gitignore

```
gedit .gitignore
```

Enregistrez et partagez vos modifications, que se passe-t-il ?

## 4 Gérer le code

Pour identifier un commit avec un nom facile à retenir, il est possible de poser un tag.  
C'est par exemple utilisé pour marquer une version donnée.

## 4.1 Des branches

Pour gérer plusieurs versions de l'ensemble du code en parallèle, Git propose de faire des «branches».

- Ajoutez une branche "nom-prenom-merge",
- dans cette branche modifiez votre article (en fin de fichier), puis validez le commit (faire au moins 2 modifications et 2 commits).
- Revenez sur la branche master,
- faites 2 modifications en début de fichiers (avec 2 commits)
- puis fusionnez la branche "nom-prenom-merge" avec master.

Observez le résultat dans l'arbre des versions.

```
git log --graph --oneline --all
```

Expliquer ce qui c'est passé.

- Ajoutez une branche "nom-prenom-rebase",
- dans cette branche modifiez votre article (en fin de fichier), puis validez votre commit (faire au moins 2 modifications et 2 commits).
- Revenez sur la branche master,
- et faites à nouveau 2 modifications en début de fichier (avec 2 commits)
- puis «rebasez».

Observez le résultat dans l'arbre des versions.

```
git log --graph --oneline --all
```

Expliquez ce qui c'est passé.

## 5 Plus loin

Expliquez chacune des commandes suivantes :

```
git log --graph --pretty=format:\
'%Cred%h%Creset -%C(yellow)%d%Creset \
%s %C(bold blue)<%an>%Creset%n' --abbrev-commit --all
git stash
git stash pop
git bisect
git staged
git unstage
git fix
git blame
git format-patch
git send-email
git archive
git revert
```