GIT

GIT est un outil de **gestion de versions**, un historique de modifications des fichiers. Cet outil est décentralisé et open source.

Pour démarrer

Il faut configurer son paramétrage avec un dépôt Git. Il faut remplacer les valeurs suivantes par vos valeurs :

- `jlord`
- `jlord@exemple.com`

```
git config --global user.name "jlord"
git config --global user.email "jlord@exemple.com"
```

Pour un confort d'utilisation, il faut créer une clé SSH et en déposer la clé publique sur Git.

```
ssh-keygen -t rsa -b 8192 -C "jlord@exemple.com"
```

Ensuite, renseigner les informations dans le projet en cours.

```
git config core.sshCommand "ssh -o IdentitiesOnly=yes -i ~/.ssh/cle-pour-git
-F /dev/null"
git remote set-url origin git@github.com:jlord/hello-world.git
```

Enfin étant donné que nous utilisons une clé spécifique, nous la renseignons dans `.shh/config`

```
# Git Account Identity
Host <jlord.github.com>
  Hostname github.com
  PreferredAuthentications publickey
  IdentityFile ~/.ssh/cle-pour-git
```

Cas pratique : récupérer un site

Simon nous propose une utilisation avec la modification d'un site depuis son ordinateur.

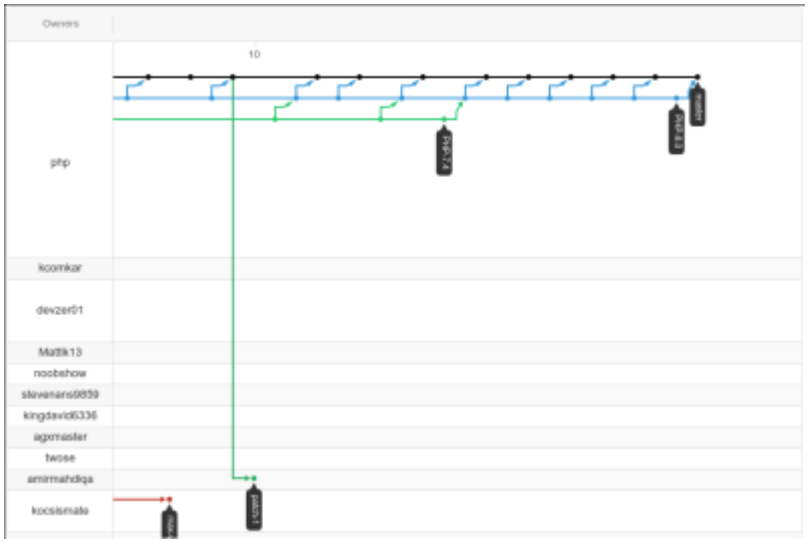
Pour participer à un dépôt Git, il faut d'abord le **cloner** (=télécharger) en local.

Un dépôt est fait de **commit**, des versions de notre code dans le temps. Un commit est créé à chaque fois qu'une fonctionnalité est implémentée et fonctionnelle.

Dans l'idéal, il faut utiliser des branches pour séparer des “objectifs”. Par exemple, si je veux mettre à

jour l'apparence de mon site fait en PHP/HTML/CSS, je vais créer une branche “moderniser-apparence”. Dans cette branche, je vais faire des **commits** des modifications de mes fichiers HTML/CSS. Un **commit** pour mettre à jour la barre de navigation, un pour mettre à jour la disposition des éléments…
Une fois que nous sommes satisfait de notre mise à jour, on **merge** (=fusionne) notre branche avec la branche principale, appelée **main** ou **master**.

Voici un exemple d'arbre du projet PHP :



Ici, je ne parlerais pas des branches que je ne connais pas suffisamment. Nous feront donc nos **commits** sur la branche main.

Voici les étapes :

1. Cloner un dépôt dans un nouveau répertoire

Le clonage d'un dépôt dans un sous-dossier

```
git clone https://github.com/jlord/hello-world.git
```

ou lorsque vous avez SSH de fonctionnel :

```
git clone git@github.com:jlord/hello-world.git
```

Ces deux exemples créeront un dossier `hello-world` à l'endroit de votre dossier de travail.

```
cd hello-world
```

2. Modifier les fichiers

Avec votre éditeur préféré (vim, vi, emacs, nano, gedit, eclipse, vs code, vscodium, sublim text …) modifiez les fichiers à votre convenance.

3. Montrer le statut de l’arbre de travail

Une fois satisfait de vos changements, regardez les modifications des fichiers que vous voulez commit :

```
git status
```

4. Mettre à jour l'index

Pour désigner les fichiers qui vont faire partie de votre mise à jour

```
git add FICHIER1[, FICHIER2...]
```

Ou pour tous les fichiers changés :

```
git add .
```

5. Enregistrer les modifications dans le dépôt

Pour enregistrer vos modifications avec une description courte.

```
git commit -m "modif NAVBAR"
```

6. Met à jour les références distantes

Une fois tout vos changements effectués, on les **push**(= téléverser / upload).

Pour le premier téléversement :

```
git push -u origin master
```

Les fois suivantes :

```
git push
```

`master` représente le nom de la branche principale.

— *Alpinux - Simon L5D 2020/12/10 19:38*

From:
https://wiki.alpinux.org/ - **Alpinux Wiki**

Permanent link:
https://wiki.alpinux.org/technique/git

Last update: **2020/12/12 21:40**