TP2 - GitHub

Le réseau de l’isat semble bloquer les connexions avec les dépôts git distant. Nous allons passer par les clés SSH. Cela évitera que git essaye d’ouvrir un navigateur pour nous demander de saisir nos identifiants à chaque fois que nous voulons interagir avec ce dépôt.

Mettre en place les clés SSH

Les clés sont stockés dans un fichier caché sur votre ordinateur situé à la racine de votre répertoire personnel.

Commencez par vérifier que vous n’avez aucune clé ssh Lancez un git bash et entrez

```
ls -a ~/.ssh
```

Les clés sont stockées sous la forme de fichier dans ce dossier.

Les clés SSH vont par deux: par exemple

- id_rsa: le fichier contenant la clé privée (sans extension)
- id_rsa.pub: le fichier contenant la clé publique (extension .pub)

Pour générer une nouvelle paire de clés, entrez

```
ssh-keygen -t ed25519 -C "your_email@example.com"
```

La commande se déclenche et vous propose plusieurs options :

- Ne spécifiez pas de chemin pour stocker les clés, tapez <Enter> pour laisser le chemin par défaut.
- Passphrase: La clé privée vous permettra de vous connecter directement sans taper de mots de passe. Si quelqu’un accède à votre ordinateur, il pourra donc se faire passer pour vous. Pour empêcher cela, vous pouvez saisir une

passphrase qui vous sera demandée dès que la clé est utilisée. Vous pouvez laissez une phrase vide.

Vous avez maintenant généré une clé publique dans un fichier id_ed25519.pub et une clé privée dans un fichier id_ed25519. Ses fichiers se trouvent dans le dossier ~/.ssh

Il faut les ajouter au gestionnaire des clés ssh, pour cela il faut tout d'abord démarrer ce gestionnaire avec la commande

```
eval "$(ssh-agent -s)"
```

Puis lui dire de considérer la clé que vous venez de générer

```
ssh-add ~/.ssh/id_ed25519
```

Enfin, il faut ajouter la clé publique dans l'interface de github, pour cela:

1. Ouvrez le fichier id_ed25519.pub, copiez la clé et rendez-vous dans github:
2. Cliquer sur votre icône en haut à droite, puis allez dans les paramètres.
3. Onglet SSH and GPG Keys
4. Bouton vert new Ssh key
5. Donner un titre qui décrit l'ordinateur sur lequel vous êtes, par exemple: "Compte iut".
6. Coller la clé à l'endroit indiqué

Synchroniser un dépôt local et un dépôt distant

Maintenant, vous pouvez synchroniser vos dépôts locaux avec les dépôts hébergés sur github.

Depuis github, allez chercher l'url d'un de vos dépôts hébergés et copiez la. Si vous n'avez aucun dépôt, vous pouvez en créer un nouveau.

Depuis votre répertoire local, dans un git bash lancez

```
git remote add origin URLDeVotreDepotEnLigne
```

Par défault, le dépôt en ligne est qualifié avec le nom 'origin'.

Il faut maintenant uploader votre code en ligne, pour cela saisissez

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

Votre dépôt local va maintenant surveiller ce dépôt en ligne, essayons de le modifier sans passer par le dépôt local:

Si on a juste des petites modifications à faire, on peut modifier les fichiers directement depuis github:

1. Modifiez le fichier index.html en cliquant sur le nom du fichier puis sur le crayon en haut à droite, ajoutez une ligne sur vos activité annexes dans votre fichier (*"je pratique les mathématiques en club, toutes les semaines." par exemple.*)
2. En bas de la page de modification, vous pouvez enregistrer les changements avec un commit. Le message par default du commit est "Update index.html" mais vous pouvez le changer. Quand vous êtes prêt, cliquez sur commit changes.
3. Vous retournez alors à la page d'accueil du dépôt.

Votre dépôt local n'a pas été modifié, si vous ouvrez le fichier avec visual studio, vous ne verrez pas la ligne que vous venez d'ajouter. Pour mettre à jour le dépôt local on utilise la commande

```
git pull
```

En lançant cette commande, git va aller chercher la version en ligne et la fusionner avec votre version locale. Il peut y avoir conflit si les deux versions ont modifié les

mêmes lignes, dans ce cas, vous devrez régler le conflit, ajouter les fichiers à la zone de préparation et faire un nouveau commit.

A chaque fois que vous voudrez mettre à jour le dépôt distant pour sauvegarder vos modifications locales, vous devrez utiliser git push. Pour cela, il faudra que votre répertoire de travail soit propre et que toutes vos modifications aient été intégrées dans un commit.

Travailler à plusieurs avec github

Pour cette partie, travaillez avec votre voisin ou par groupe de 3, inutile que chacun d'entre vous ne suivent les étapes.

On peut utiliser git de son côté lorsqu'on travaille sur un gros projet mais git est surtout utilisé lorsqu'il s'agit de travailler à plusieurs. Pour illustrer cela, vous allez travailler sur un dépôt commun avec vos voisins. Pour autoriser un autre utilisateur à modifier votre code, vous devez modifier les autorisations de votre dépôt:

1. Aller dans les paramètres de votre dépôt distant: l'onglet "settings" avec la roue dentée.
2. Accéder au panneau "collaborators"
3. Cliquer sur add people et tapez le nom d'utilisateur de vos voisins et ajoutez les.
4. Vos voisins doivent maintenant aller accepter l'invitation en cliquant sur le lien qu'ils ont reçu par mail.
5. Vos voisins ont accès à votre dépôt et peuvent envoyer du code.

Lorsque vous voulez travailler sur un dépôt distant, copiez son url et placez-vous dans un dossier vide. Vous pouvez alors lancer

```
git clone URLduDepotDistant
```

Une fois que les fichiers ont été téléchargés, vous pouvez travailler dessus et les modifier. Ajouter une ligne au fichier index.html pour ajouter une facette de la personnalité de votre voisin qu'il n'a pas précisé sur sa page, puis faite un commit avec cette modification. Ensuite envoyer votre code sur le dépôt distant de votre voisin avec

```
git push
```

Voilà, le dépôt situé sur le compte de votre voisin a été modifié. S'il veut travailler sur la dernière version, il faudra qu'il lance un `git pull` pour télécharger la version la plus récente, qu'il règle les conflits s'il y en a, et fasse un commit, puis qu'il mette à jour le dépôt distant avec un `git push`

- Arrangez-vous avec votre voisin pour modifier la même ligne du fichier index.html. Celui qui fera son `git push` en premier n'aura pas à régler de conflit. Lorsque le premier aura fait son `git push`, les autres ne pourront plus faire de `git push`, il faudra d'abord effectuer un `git pull` et régler les conflits, puis faire un commit et un nouveau `git push` . Enfin, tout le monde pourra faire un `git pull` pour avoir la dernière version.

Mettre en ligne sa page web

Commencez par supprimer vos voisins de la liste de vos collaborateurs, pour ne pas avoir de mauvaise surprise.

Pour héberger sa page web sur github, il faut créer un dépôt intitulé "nomUtilisateur.github.io" et il faut que ce dépôt possède un fichier index.html.

GitHub annonce un délai de 20 minutes pour mettre à jour le site web, une fois passé ce délai, vous pourrez accéder à la page depuis n'importe où en tapant "https://nomUtilisateur.github.io" dans un navigateur.

- Ajoutez une image à votre site web. Téléchargez une image, ajoutez là à votre dépôt local, modifiez votre fichier html puis publiez les modifications.

Profitez de cet endroit pour montrer vos compétences techniques en html, css, javascript. Mettez un lien vers votre cv, vos projets personnels et vos passions et ajoutez le lien en signature de vos mails. Bref vous pouvez en faire ce que vous voulez.

Vous pouvez héberger plusieurs dépôt sur internet. Il suffit depuis un dépôt d'aller dans les paramètres, panneau 'pages' à droites, puis sélectionner la branche, votre

site sera publié à l’adresse https://nomUtilisateur.github.io/nomDuDepot

Reprenez le site que vous avez développé en cours de web, ou bien en saé et hébergez le en ligne.