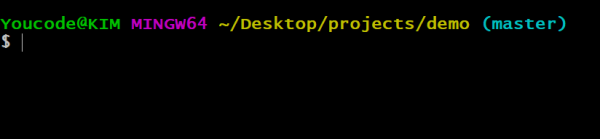
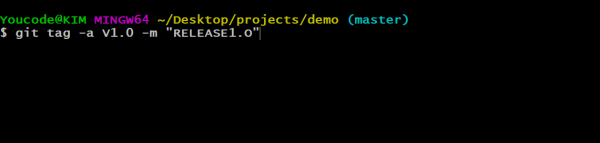
#1: First Step : Tagging

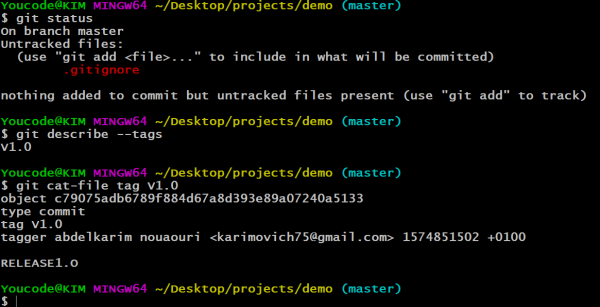
⇒ Déplacez-vous sur la branche Principale



⇒ Créez un TAG avec un nom V1.0 et un commentaire ‘ RELEASE 1.0 ‘

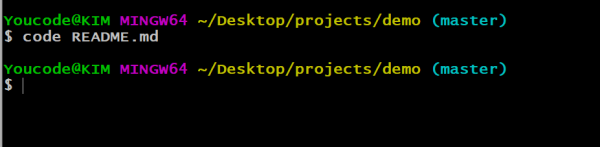


⇒ Affichez les informations sur le TAG

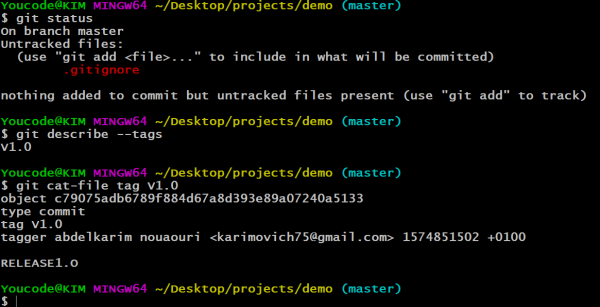


#2: Second Step: Stashing and saving work in Progress

⇒ Modifiez le fichier README.md , ajoutez une ligne



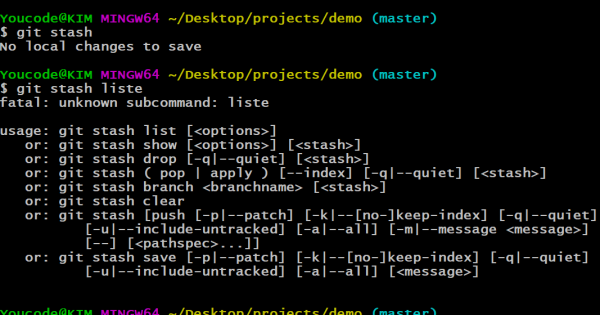
⇒ Explication du TAG: Git tag nous offre la possibilité étiqueter un certain état d'historique. Par exemple : Pouvoir étiqueter les état de publications ou release par 0.1 - 0.2 - 0.3 pour pouvoir se repérer plus tard.



⇒ definition GIT STASH: Git stash est une commande de Git qui permet de "mettre de côté" des modifications. Cette fonctionnalité est très pratique, notamment lorsque le développeur a besoin de changer de branche rapidement. Cette fonctionnalité lui permet de ne pas se presser à effectuer un commit bâclé pour pouvoir changer de branche.

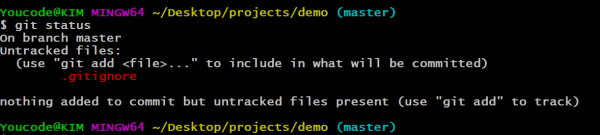
Git stash va récupérer les modifications en cours et les enregistrer dans un conteneur qu'on appellera un stash.

⇒ Tapez la commande git stash list

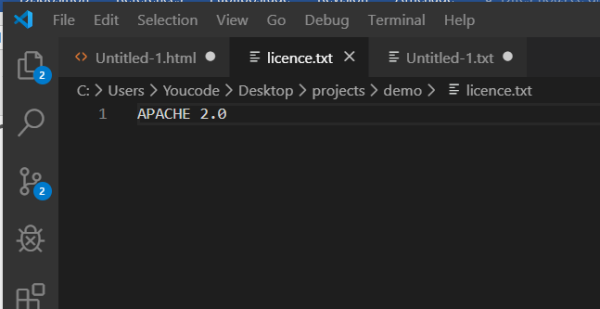


⇒ Exécutez la commande git status ?

⇒ Qu’est-ce que vous constatez ?



⇒ Modifiez le fichier « Licence.txt », ajoutez la ligne « APACHE 2.0 »



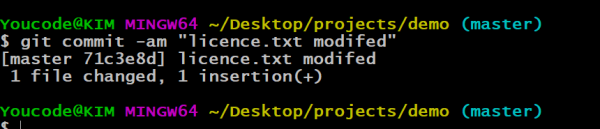
⇒ git stash list



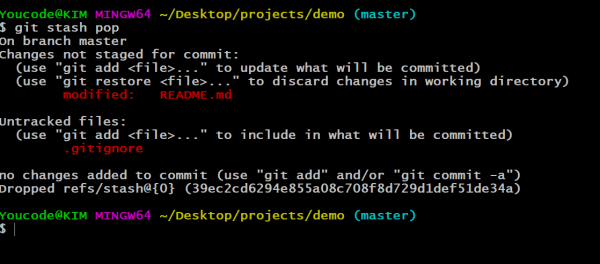
⇒ Git stash est une commande de Git qui permet de mettre en backstage des modifications afin d'éviter d'effectuer un commit bâclé pour pouvoir changer de branche.

Git stash va récupérer les modifications en cours et les enregistrer dans un conteneur qu'on appellera un stash.

⇒ Faites stagging et le commit en une seule ligne



⇒ Exécutez la commande « git stash pop »



#3: third step: Voyage sur Github, Local Repo to github Repo

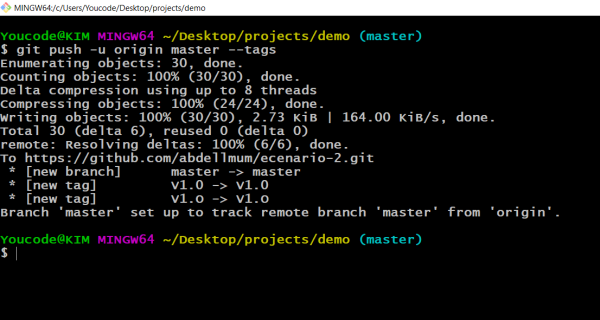
⇒ Créez un repo github public sans ajouter le fichier README.md

⇒ Créez le remote en https

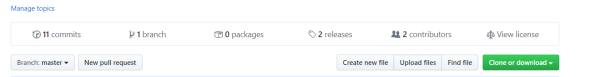
⇒ Examinez le remote



⇒ Pushez le tous à travers la commande : git push –u origin master - -tags



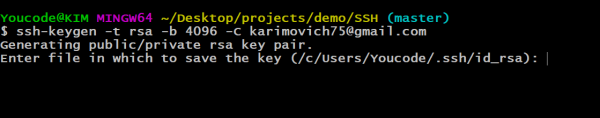
⇒ Sur Github Vérifiez la liste des commits , les branches , les releases et les tags



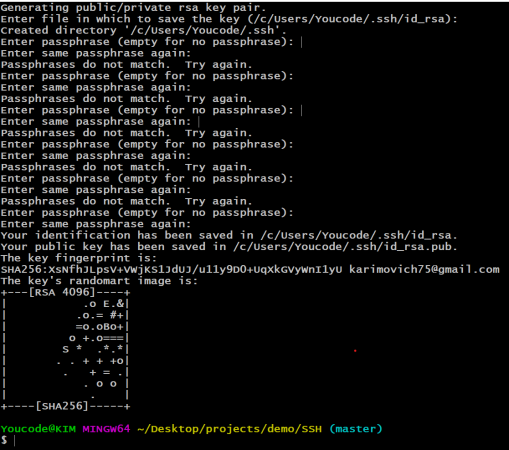
#4: Fourth Step: Mini challenge (optionnel)

⇒ Déplacez-vous dans le fichier .SSH

⇒ Maintenant à vos mains : Créez une authentification sécurisé SSH entre votre repo local et votre repo distant (aide : Ssh –keygen-t rsa –C ' email’)



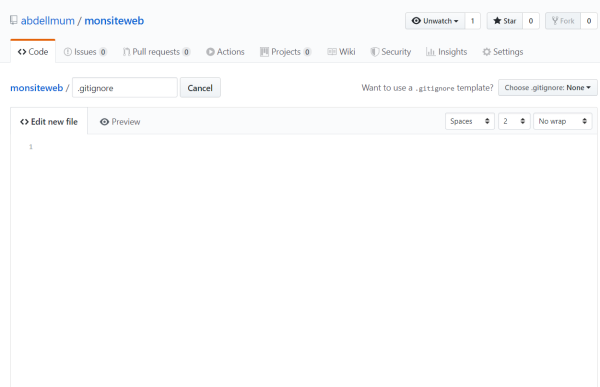
⇒ SSH keygen password reat



#5: Fifth Step: Création d’une local copy

⇒ Sur github créez un autre repo nommé ( Monsiteweb)

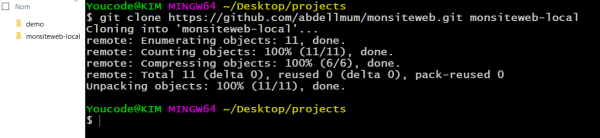
⇒ Ajoutez à l’arboresence toujours sur github le fichier .gitignore et un fichier licence.txt ‘APACHE 2.0 ‘



⇒ Déplacez-vous dans le répertoire projets sous GIT

⇒ Créez un clone github vers le local sous le nom (Monsiteweb-local)

⇒ Vérifiez si le clone est créé.

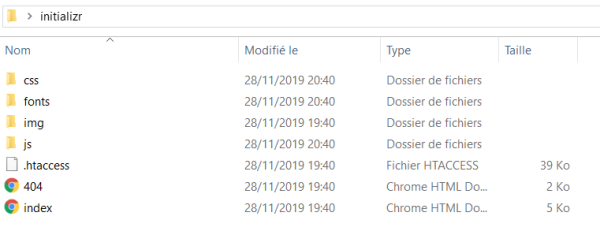


#6: sixth step : Sending the website

⇒ Télécharger le site web depuis le lien suivant :<http://www.initializr.com/>

⇒ Sur le site telecharger un site bootstrap avec le fichier .htaccess et le fichier 404.html

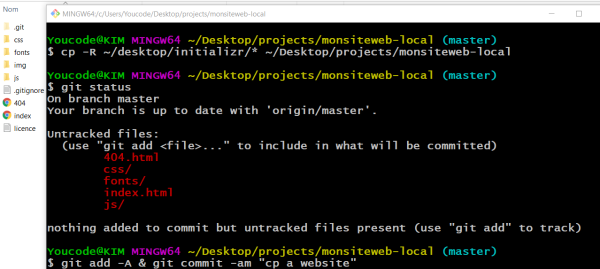
⇒ Analysez l’arboresence



⇒ Copiez le site télécharger dans votre repo local à travers une seule commande :

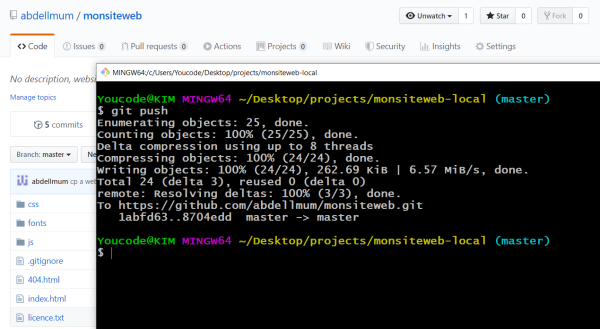
Git status

⇒ Faites le stagging et le commit en une seule ligne



⇒ Faite le push à github

⇒ Vérifiez l’existence du site local sur votre repo github



#7: seventh step: Fetch and pull

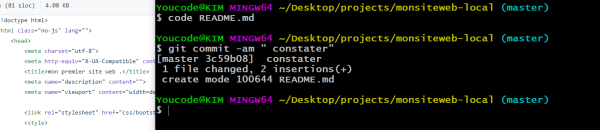
⇒ Sur github éditez le fichier Index.html, sur la balise <title> </title> ajoutez le titre , mon premier site web .

⇒ Faites le commit sur github

⇒ Sur git et sur le repo local , Editez le fichier README.md

⇒ Faites le stagging et le commit en une seule ligne

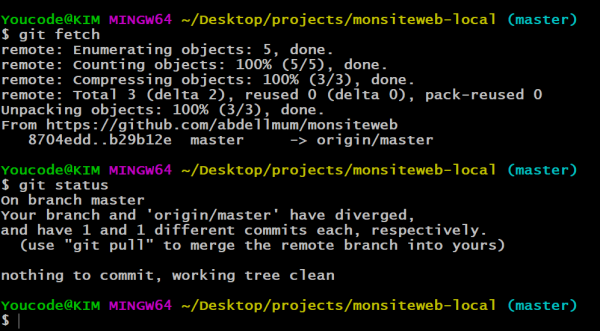
⇒ Qu’est-ce que vous constatez ?



⇒ Executez la commande « git fetch »

⇒ Git status

⇒ Expliquez ?



⇒ Git pull

⇒ Git push

⇒ Verifiez les commit sur github

