Une fois que vous vous êtes placés dans votre nouveau dossier grâce à la commandecd de votre Terminal, créez un nouveau dossier 'monPremierRepo' en lançant la commande suivante :

mkdir monPremierRepo

 Pour activer un dossier comme repository Git, il suffit de se placer dans ce dossier avec le Terminal puis d'utiliser la commande **git init**.

 Pour gérer un repository, Git génère un index de tous les fichiers dont il doit faire le suivi. Lorsque vous créez un fichier dans un repository, vous devez donc l'ajouter à l'index Git à l'aide de la commande **git add nomDeVotreFichier.extension**. Par exemple :

git add checklist-vacances.md

Pour gagner du temps, vous pouvez ajouter tous les fichiers dans le répertoire courant en tapant

git add .

 Lorsque vous modifiez votre repository, vous devez demander à Git d'enregistrer vos modifications en faisant un **git commit**. L'option-m vous permet de lui envoyer un message décrivant les modifications effectuées, ce qui s'avèrera très utile pour vous par la suite, you'll see! :)Par exemple :

git commit -m "Ajouté ma checklist-vacances.md (woohoo!)"

Comment vous y retrouver dans l'historique de vos commits ?

Grâce à la commande  git log qui vous affiche la liste de tous les commits que vous avez réalisés !

Le sens de lecture du log est le suivant : on part du commit le plus récent en haut de la liste, au commit le plus vieux en bas de la liste.

Et bien, si vous ne faites que mettre à jour un fichier que vous aviez déjà ajouté à l'index, vous pouvez condenser ces deux étapes de la façon suivante :

git commit -a -m "Ajouté itinéraire dans checklist-vacances.md"

L'option   -a demande à Git de mettre à jour les fichiers déjà existants dans son index.

Lorsque vous effectuez une série de commits sur un projet, il peut vous arriver de vouloir remonter dans le temps à la recherche d'erreurs éventuelles par exemple. Pour vous positionner sur un commit donné de votre historique, il vous suffit d'utiliser la commande **git checkout** de la façon suivante :

git checkout SHADuCommit

Pour revenir à votre branche principale (au commit le plus récent), on utilise la même commande :

git checkout master

Je vous présente une de ces options : vous pouvez "revert" un commit, c'est-à-dire créer un nouveau commit qui fait l'inverse du précédent, avec la commande suivante :

git revert SHADuCommit

Sinon, si vous voulez simplement modifier le message de votre dernier commit, vous pouvez utiliser la commande suivante :

git commit --amend -m "Votre nouveau message"

... mais attention, vous ne pouvez modifier ce message que si vous n'avez pas encore pushé votre commit sur l'origine !

Je n'ai pas encore fait mon nouveau commit, mais je veux annuler tous les changements que je n'ai pas encore commités. Possible ?

 Possible avec un reset !

git reset --hard‌