Démarrer avec le gestionnaire de version Git

Etape 1 – Créer un dépôt initial.

Git gère vos informations dans un dépôt (repository), qui correspond en réalité à un répertoire caché de votre répertoire de travail (celui qui contient les fichiers dont vous voulez conserver différentes versions). Ce répertoire ne sera géré que par git, vous ne devez donc ne pas vous en préoccuper.
Pour initialiser le dépôt, vous utilisez la commande :

```
git init
```

Etape 2 – Configurer votre environnement.

Lorsque git enregistre les informations que vous lui soumettez, git ajoute le nom de la personne qui demande l'enregistrement (la sauvegarde). Cela permet, lorsqu'on travaille dans un environnement multi-collaborateurs, de retrouver qui a enregistré quoi.
Pour configurer votre environnement, vous utilisez les 2 (obligatoirement) commandes (notez l'utilisation de double-tiret) :

```
git config --global user.name VotreNom
git config --global user.email VotreAdresseMail
```

Etape 3 – Préparer votre prochaine sauvegarde.

Vous devez indiquer à git quel est l'ensemble des changements dans vos fichiers que vous voulez qu'il conserve pour la prochaine fois que vous demanderez la sauvegarde.
Pour cela, vous indiquez à git de placer vos changements dans une zone temporaire (qu'on appelle l'index, et dont le contenu est conservé jusqu'à la prochaine sauvegarde), grâce à la commande suivante :

```
git add NomDuFichier
```

Attention, vous aurez à utiliser cette commande chaque fois que vous voulez conserver des changements dans vos fichiers. Si vous n'êtes pas explicite, git ne saura pas quoi enregistrer. En réalité, git va conserver la totalité du fichier, comme si il avait pris une photo des données dans le fichier, à l'instant où vous avez demandé "`add`".

Etape 4 – Savoir ce qui a été enregistré dans l'index.

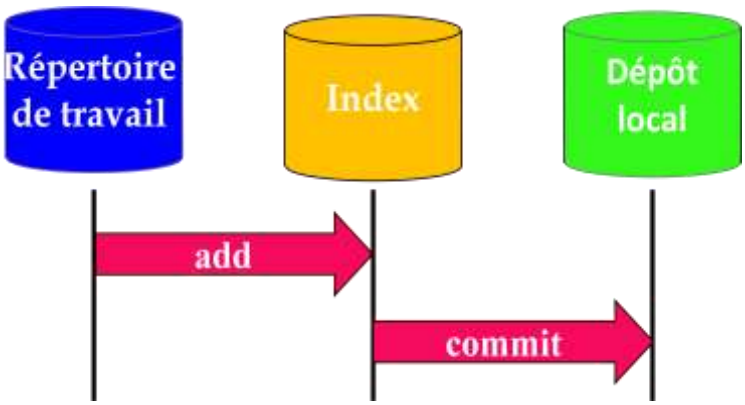
La commande `git status` affiche la situation de chacun des fichiers de votre répertoire de travail (et fichiers des sous-répertoires éventuellement), et vous donne aussi des indices sur ce que vous pourriez faire pour faire gérer vos données.

Etape 5 – Enregistrer vos informations dans le dépôt.

Cela s'appelle "faire un commit". Lorsque vous enregistrez un ensemble de changements (contenus dans vos fichiers), git enregistre ce qui est dans l'index, et vous demande de décrire en quoi concernent les changements. Il vous faut donc fournir, soit un fichier descriptif, soit, au minimum, un message court qui décrit votre dernière contribution.
Par exemple : "*Ajout d'un test pour la fonction getValue : cas où l'input est 3*"
La commande qui permet de "committer" est

```
git commit -m "Message qui décrit les changements"
```

L'option `-m` est celle qui permet d'écrire un message court. Si vous préférez être loquace, sans cette option, git lancera un éditeur de texte pour vous permettre de vous exprimer.



Etape 6 – Connaître l'historique des commits de votre projet.

La commande `git log` affiche, en ordre chronologique inversé, chaque commit réalisé sur le dépôt, l'heure à laquelle il a été créé, le nom de la personne qui l'a effectué et la première ligne du message de description de ce commit.

Etape 7 – Partager votre travail.

Vous pouvez envoyer vos informations sur un dépôt localisé sur une autre machine, afin que d'autres utilisateurs prennent connaissance de votre travail.
Pour cela, vous pouvez vous créer un dépôt sur GitHub, et en retenir l'adresse. Ainsi, si votre identifiant sur GitHub est *Lolo* et que votre dépôt sur GitHub se nomme *TD8*, alors son URL sera *https://github.com/Lolo/TD8.git*
Dans un premier temps, vous devez associer votre dépôt local avec celui qui a été créé sous GitHub. Pour cela, vous utilisez la commande

```
git remote add origin https://github.com/Lolo/TD8.git
```

Ensuite pour envoyer vos informations sur GitHub, vous utilisez

```
git push origin master
```

Pour récupérer des informations à partir d'un dépôt distant, il faut aussi, dans un premier temps, associer un (second) dépôt local au dépôt GitHub, puis utiliser la commande :

```
git pull origin master
```

pour télécharger en local les informations distantes.