```
Installer GIT

Créer un dossier manuellement ou bien avec Visual Studio
Créer des fichiers dans ce dossier

Accéder à ce dossier grâce à la console
-------------------------------------------------------------------------------
On va initialiser un nouveau dépôt Git
console> git init
-------------------------------------------------------------------------------
On affiche le statut actuel de la branche
console> git status
-------------------------------------------------------------------------------
On va ajouter les fichiers afin qu'ils soient traqués par Git
console> git add .
-------------------------------------------------------------------------------
On va écrire un message qui va décrire le commit (les modifications effectuées)
console> git commit -m "Texte de description du commit"
-------------------------------------------------------------------------------
On affiche la liste des branches
console> git branch
-------------------------------------------------------------------------------
Effectuer une petite modification dans un fichier du répertoire
console> git add .
console> git commit -m "deuxième modification"
console> git log
-------------------------------------------------------------------------------
On revient en arrière sur un commit. Taper la commande suivante avec l'id de l'avant
dernier commit

console> git reset --hard IDCOMMIT
-------------------------------------------------------------------------------
On peut revenir au dernier push

console> git checkout -- .
-------------------------------------------------------------------------------
On va créer une nouvelle branche pour une nouvelle fonctionnalité

console> git checkout -b new-feature
-------------------------------------------------------------------------------
On regarde la liste des branches
console> git branch

En tapant "git log" on remarque que tous les précédents commits existent dans
la nouvelle branche

console> git log
-------------------------------------------------------------------------------
On ajoute une nouvelle fonction (créer un fichier style.css)
console> git add .
console> git commit -m "css added"
console> git log
-------------------------------------------------------------------------------
On va fusionner dans master les modifications
console> git checkout master
console> git merge new-feature
console> git log
-------------------------------------------------------------------------------
On peut supprimer la branche "new-feature".
Le travail qui a été mergé sur "master" ne sera pas supprimé

console> git branch
console> git branch -D new-feature
console> git branch
console> git log
-------------------------------------------------------------------------------
// On crée un nouvelle branche
console> git checkout -b new-feature

// On modifie le fichier en changeant la première ligne par exemple.
console> git add .
```

```
console> git commit -m "modification-feature"

// On va simuler qu'on a modifié la même ligne sur master
console> git checkout master

// On modifie le même fichier à la même ligne puis on refait un commit
console> git add .
console> git commit -m "modification-master"

// Que se passe t'il si on essaye de réaliser le merge ? Conflit détecté !
console> git merge new-feature

// Il apparait nécessaire de choisir l'une ou l'autre des modifications et
// de réaliser un commit sur master avant de réaliser la fusion
--------------------------------------------------------------------------------
// Utilisation avec GitHub
// Créer un nouveau dossier "github" dans l'endroit souhaité pour le test.
// Ouvrir Visual Studio Code, "OUVRIR UN DOSSIER" et sélectionner le nouveau dossier
// Créer un fichier "index.html"
console> git init
console> git add .
console> git commit -m "start"

// On peut voir la liste des branches
console> git branch

// On va sur github.com pour créer un compte public
// On crée aussi un repository intitulé "git"
// On nous propose d'initialiser un git local ou bien d'ajouter une origine
// Il suffit de copier coller les instructions fournies.

// On va ajouter une origine externe à notre repository local.
console> git remote add origin url-de-votre-repository

// On affiche la liste des connexions pour le repository local
console> git remote
console> git remote -v

// Si on vérifie dans github, on ne voit pas les branches et les fichiers.
// En effet, nous n'avons pas encore fait le "push" qui consiste à envoyer
// les fichiers vers le server repository remote origin.

// On va envoyer les informations locales vers le serveur remote (ici Github)
console> git push origin master

// En vérifiant sur GitHub, on remarque que les nouvelles informations
// ont bien été envoyées vers le Serveur distant GitHub

// On peut consulter les informations sur les branches
console> git branch
console> git remote
console> git branch -r
--------------------------------------------------------------------------------
// On veut maintenant cloner le projet. Il s'agit de cloner un repository distant
// vers l'ordinateur local et ainsi avoir un repository clôné en local.

// Créer un nouveau dossier sur l'ordinateur et ouvrez le avec visual studio code
console> git clone url-de-votre-repository

// Si on affiche le statut de ce dossier, une erreur se produit. En effet, il faut
// être à l'intérieur du dossier du repository local.
console> git status

// Entrer dans le dossier du repository.
console> git status

// Ajouter un nouveau fichier "style.css" et le traquer avec git
console> git add .
console> git commit -m "adding css"

// Si on tape git remote, on voit que toutes les connexions sont déjà réalisées
```

```
143   // En effet quand on a cloné, on a bénéficié de toute la structure du "code history"
144   // et des connexions aux branches.
145   console> git remote
146
147   // On remarque que la connexion HEAD est déjà préconfigurée et connectée.
148   console> git branch -r
149
150   // On peut utiliser le raccourci pour "pusher" vers la branche connectée
151   console> git push
152   ------------------------------------------------------------------------------------
153   // En regardant sur GitHub, on voit que la nouvelle fonctionnalité a bien été
154   // envoyée. Mais attention, sur le clone du repository, l'utilisateur n'a pas
155   // la nouvelle fonctionnalité dans son repository local ! Ses fichiers sont anciens
156
157   // Afin de vérifier s'il y a une nouvelle fonctionnalité sur le serveur
158   // Les fichiers du repository local ne seront pas modifiés
159   console> git fetch
160
161   // On peut merger les informations en provenance du serveur distant. Cela permet de
162   // mettre à jour les fichiers du repository local à partir des dernières versions
163   // disponibles sur le serveur.
164   console> git merge origin/master
165
166   // On peut réaliser ces deux opérations à l'aide d'une seule commande
167   console> git pull
168   ------------------------------------------------------------------------------------
169   // On peut supprimer un repository sur Github en allant dans "Settings"
170   // Puis en bas dans "Danger Zone" : Delete this repository.
171   // Il faut réecrire le nom du repository afin de le supprimer
172
173   // Si on vérifie les connexions locales, on remarque qu'elles existent toujours
174   console> git remote
175
176   // Pour supprimer la connexion avec origin
177   console> git remote rm origin
178   console> git remote
179   ------------------------------------------------------------------------------------
180
```