

Gérer ses ressources à travers GIT

COMMANDE GIT

CONTENU

| | |
|---------------------------------------------------------------------|---|
| Outil à utiliser..... | 1 |
| Aides | 1 |
| Initialisation d'un répertoire GIT..... | 1 |
| Suivi des modifications | 1 |
| Commit des modifications sur branche locale..... | 1 |
| Annulation des modifications en cours | 1 |
| Récupérations des modifications d'un repository distant..... | 2 |
| Dépôt des modifications sur un repository distant | 2 |
| Création d'une branche | 2 |
| Naviguer d'une branche à l'autre..... | 2 |
| Merge d'une branche à l'autre | 2 |
| Exercices | 3 |
| GIT - Installation des outils | 3 |
| GIT - Création d'un repository personnel | 3 |
| GIT - Récupération du cours | 4 |
| GIT - Création du fichier README.md | 5 |
| Github - Création d'un compte et repository github | 6 |
| Git - travail collaboratif (travail à plusieurs) | 7 |
| Git - gestion des conflits automatiques (travail à plusieurs) | 7 |
| Git - gestion des conflits manuels..... | 7 |
| Schéma récapitulatif | 8 |

OUTIL A UTILISER

Git for windows disponible à cette adresse [Git for Windows](#)

Github desktop disponible à cette adresse [GitHub Desktop | Simple collaboration from your desktop](#)

AIDES

`git` (*liste Les commandes disponibles*)

`git maCommande --help` (*exemple : `git add --help`*)

INITIALISATION D'UN REPERTOIRE GIT

`git init` (*dans Le répertoire source vide*)

`git clone` uneAdresse (*un sous-répertoire sera créé*)

SUIVI DES MODIFICATIONS

`git status`

COMMIT DES MODIFICATIONS SUR BRANCHE LOCALE

`git add .` (*indexation de tous Les fichiers*)

`git commit -a -m "Description du commit"`

ANNULATION DES MODIFICATIONS EN COURS

`git reset -hard`

RECUPERATIONS DES MODIFICATIONS D'UN REPOSITORY DISTANT

`git fetch` (*récupère les informations de modification en ligne*)

`git pull` (*applique les modifications préalablement récupérées*)

DEPOT DES MODIFICATIONS SUR UN REPOSITORY DISTANT

`git push` (*à ne faire qu'après avoir fait un fetch/pull et commit*)

CREATION D'UNE BRANCHE

`git checkout -b "NouvelleBranche"`

NAVIGUER D'UNE BRANCHE A L'AUTRE

`git checkout "nomDeLaBrancheCible"` (ex. `git checkout master`)

MERGE D'UNE BRANCHE A L'AUTRE

Depuis la branche cible

`git merge --no-ff "BrancheSource"`

EXERCICES

GIT - INSTALLATION DES OUTILS

- Installer GIT for Windows <https://gitforwindows.org/>
- Installer github desktop <https://desktop.github.com/>

GIT - CREATION D'UN REPOSITORY PERSONNEL

- Quelques commandes qui vous seront utiles pour cet exercice :
 - **ls** : liste les fichiers du répertoire courant
 - **pwd** : affiche le répertoire courant
 - **cd** : permet de naviguer dans un sous dossier (**cd monSousDossier**), de remonter au dossier parent (**cd ..**) ou d'aller dans un répertoire spécifique (**cd /c/Users/ameistertzheim/Documents/Github** ou **cd c:\Users\ameistertzheim\Documents\Github**)
 - **mkdir** : création d'un nouveau dossier (**mkdir monSousDossier**)
 - **touch** : création d'un nouveau fichier (ex : **touch readMe.md**)
 - **start .** : ouverture d'un explorateur de fichier dans le dossier courant
 - **rm** : suppression d'un fichier (ex : **rm readMe.md**)
 - **rmdir** : suppression d'un dossier (ex : **rmdir Algo**)

- Créez un répertoire local qui contiendra tous les repository futurs (ex. **C:\Users\ameistertzheim\Documents\Github**)
- Lancez une console git (git Bash ou git Cmd) puis naviguez vers le répertoire créé

```
Adeline@CRM-UC-3620 MINGW64 ~
$ cd Documents/

Adeline@CRM-UC-3620 MINGW64 ~/Documents
$ cd Github/

Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github
$ |
```

- Créez un sous-répertoire "ABCDEDEV_2310 ", suivi de vos initiales par exemple ABCDEV_2306_MaD, en ligne de commande

```
Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github
$ mkdir ABCDEV_2310_MaD
```

- Naviguez dans ce répertoire

```
Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github
$ cd ABCDEV_2310_MaD/

Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github/ABCDEDEV_2310_MaD
$ |
```

- Initialisez un repository git

```
Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github/ABCDEDEV_2310_MaD
$ git init
Initialized empty Git repository in C:/Users/Adeline/Documents/Github/ABCDEDEV_2310_MaD/.git/

Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github/ABCDEDEV_2310_MaD (master)
$ |
```

- Ouvrez l'explorateur de fichier et regardez l'état du dossier

```
Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github/ABCDEV_2310_MaD (master)
$ start .
```

Ce PC > Documents > Github > ABCDEV_2310_MaD

| Nom | Modifié le |
|----------------------------------------------------------------------------------------|------------------|
|  .git | 26/10/2023 11:35 |

- Créez les différents dossiers (Algo, Objet, Pseudo-code, Diagramme_UML et java) et les différents fichier readMe.md afin d'obtenir l'arborescence suivante :

- ABCDEV_2310_MaD
 - Algo
 - Pseudo_code
 - Java
 - readMe.md
 - Objet
 - Diagramme_UML
 - Java
 - readMe.md
 - ReadMe.md

Exemple pour créer le dossier Algo et le fichier readMe.md :

```
Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github/ABCDEV_2310_MaD (master)
$ mkdir Algo






Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github/ABCDEV_2310_MaD (master)
$ touch readme.md
```

GIT - RECUPERATION DU COURS

- Lancez une console git puis naviguez jusqu'au répertoire contenant les repositories (Ex. C:\Users\ameistertzheim\Documents\Github)
- Clonez le repository en ligne suivant : https://github.com/AdelineMeister/ABCDEV_2310_MaD.git

```
Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github
$ git clone https://github.com/AdelineMeister/ABCDEV_2310_MaD.git
Cloning into 'ABCDEV_2310_MaD'...
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 7 (delta 0), reused 7 (delta 0), pack-reused 0
Receiving objects: 100% (7/7), 427.28 KiB | 2.01 MiB/s, done.
```

- Ouvrez l'explorateur de fichier et regardez l'état du dossier créé

| Nom | Modifié le | Type | Taille |
|-----------------------------------------------------------------------------------------------|------------------|-----------------------|--------|
|  .git | 26/10/2023 11:49 | Dossier de fichiers | |
|  Algo | 26/10/2023 11:49 | Dossier de fichiers | |
|  Git | 26/10/2023 11:49 | Dossier de fichiers | |
|  Objet | 26/10/2023 11:49 | Dossier de fichiers | |
|  readMe.md | 26/10/2023 11:49 | Fichier source Mar... | 1 Ko |

GIT - CREATION DU FICHIER README.MD

- Créez un fichier appelé README.md dans votre repository personnel.
 - Pour info : l'extension ".md" est utilisée pour les fichiers en langage "markdown" (cf. [Wikipedia](#)) et peuvent être modifiés par un simple éditeur de texte
 - Modifiez le contenu du fichier README.md par un texte que l'on modifiera plus tard, puis sauvegarder

Exemple :

```
# Mon répertoire personnel
=====
Ceci est mon répertoire personnel
```

- Regardez ce que la commande "git status" vous donne

```
Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github/ABCDEV_2310_MaD (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  readme.md

nothing added to commit but untracked files present (use "git add" to track)
```

- Effectuez votre premier commit avec comme description "Commit initial"

```
Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github/ABCDEV_2310_MaD (master)
$ git add .

Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github/ABCDEV_2310_MaD (master)
$ git commit -a -m "Cimmit initial"
[master (root-commit) 2ade4b6] Cimmit initial
1 file changed, 7 insertions(+)
create mode 100644 readme.md
```

- Regardez ce que "git status" vous donne

```
Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github/ABCDEV_2310_MaD (master)
$ git status
On branch master
nothing to commit, working tree clean
```

- Modifiez le fichier README.md pour y inclure votre Nom, prénom, puis sauvegardez les modifications

- Regardez ce que la commande "git status" vous donne

```
Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github/ABCDEV_2310_MaD (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   readme.md

no changes added to commit (use "git add" and/or "git commit -a")
```

- Effectuez votre second commit (attention à bien indexer les modifications via la commande git add)
- Effectuez une dernière modification à README.md que l'on annulera plus tard

- Effectuez un git status, puis annulez vos modifications via git, et regardez le résultat

```
Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github/ABCDEV_2310_MaD (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   readme.md

no changes added to commit (use "git add" and/or "git commit -a")

Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github/ABCDEV_2310_MaD (master)
$ git restore readme.md

Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github/ABCDEV_2310_MaD (master)
$ git status
On branch master
nothing to commit, working tree clean

Adeline@CRM-UC-3620 MINGW64 ~/Documents/Github/ABCDEV_2310_MaD (master)
$ |
```

GITHUB - CREATION D'UN COMPTE ET REPOSITORY GITHUB

- Sur github.com, créez-vous un compte personnel gratuit, puis créez un nouveau repository en ligne appelé "ABC_DEV_2310 ", **sans cocher l'option "d'initialisation d'un fichier README.md"**
- Depuis la console git en local naviguer jusqu'à votre repository personnel
- Puis effectuez les opérations telles qu'affichées par github sous le titre "...or push an existing repository from the command line", à savoir :


```
git remote add origin
https://github.com/<votreNomDUtilisateurGithub>/ABCDEV_2310.git
git branch -M main
git push -u origin main
```
- Git devrait vous demander votre login et mot de passe github. Renseignez ceux du compte créé
- Regardez le résultat sur github
- Effectuez une modification de readme.md, sauvegardez, et faites un commit de cette modification
- Vérifiez l'état en local et sur github :
 - Le commit existe en local mais pas encore sur github

GIT - TRAVAIL COLLABORATIF (TRAVAIL A PLUSIEURS 😊)

- Clonez le repository github d'un de vos collègues dans un nouveau repository local
- A partir de votre repository personnel, effectuez un "dépôt" de vos modifications de l'exercice précédent sur github et visualisez le résultat en ligne
- Lorsque votre collègue en aura fait de même, dans le clone de son repository sur votre machine, effectuez une récupération des modifications.
- Dans le clone du repository de votre collègue sur votre machine, modifiez son README.md, et effectuez un commit
 - La modification en local est possible
- Essayez d'effectuer un dépôt de ce repository
 - Cette opération est impossible, vous n'avez pas les droits
 - Si la lecture est publique, la modification est privée
- Sur github, dans les paramètres du repository, ajoutez votre collègue en tant que collaborateur, et acceptez sa demande
- Tentez d'effectuer à nouveau un dépôt du clone du repository de votre collègue : la modification est maintenant possible
- Récupérez les modifications de votre propre repository : les modifications de votre collègue ont bien été appliquées

GIT - GESTION DES CONFLITS AUTOMATIQUES (TRAVAIL A PLUSIEURS)

- Sur votre machine et sur celle de votre collègue, effectuez deux modifications compatibles du fichier README.md (exemple : modification de lignes séparées), rajoutez un fichier différent et effectuez un commit en local sur les 2 machines
- Sur l'une des 2 machines, effectuez un dépôt en ligne sur github
- Sur la seconde machine, récupérez les modifications : normalement, les conflits sont automatiquement résolus, vous avez bien à la fois vos modifications et celles de votre collègue sur votre machine locale.
- Effectuez un dépôt à partir de la seconde machine
 - Regardez le résultat sur github, notamment sur l'historique des commits
- Enfin, effectuez un rafraîchissement sur la 1^{ère} machine : vous avez alors tous les deux le même code.

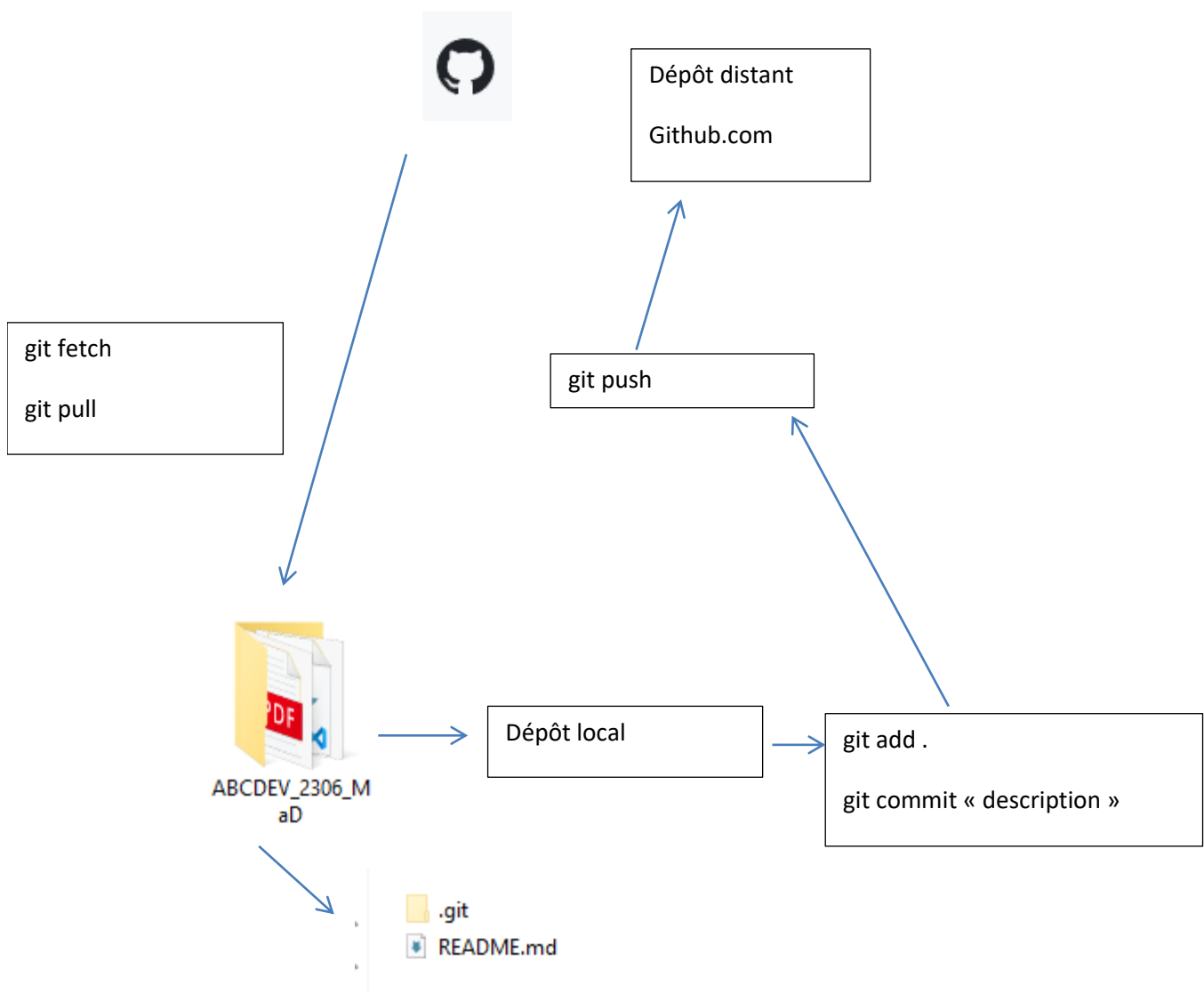
GIT - GESTION DES CONFLITS MANUELS

- Sur votre machine et sur celle de votre collègue, effectuez deux modifications incompatibles du fichier README.md (modification d'une même ligne d'un même fichier, commencez simplement) et effectuez un commit en local sur les 2 machines
- Sur l'une des 2 machines, effectuez un dépôt en ligne sur github
- Sur la seconde machine, récupérez les modifications : normalement, git devrait vous informer d'un conflit qu'il ne sait pas gérer, ce qui est normal.
- Retournez dans l'explorateur de fichier vers votre repository
 - Utilisez Github Desktop pour visualiser les lignes en conflits
 - L'outil qui apparait permet de naviguer entre les lignes en conflit, puis pour chacun, de spécifier si l'on souhaite prendre l'option de droite, de gauche, ou les 2. Le résultat final apparaissant en bas de la fenêtre.

- Il est possible de modifier directement la partie basse de la fenêtre
- Une fois les conflits résolus, vous pouvez fermer la fenêtre et valider les conflits, puis effectuer un commit qui comprendra la résolution du conflit
- Effectuez plusieurs fois cette opération de gestion de conflit pour prendre en main l'outil. Tentez la modification de plusieurs lignes, rajoutez des fichiers, supprimez des fichiers modifiés par l'autre etc. Ce que vous faites ici sur des exemples simples vous facilitera la vie plus tard. Il est important de savoir manipuler l'outil de merge pour que plus tard, la complexité de la résolution soit dans l'écriture du code résultant, et non pas dans la manipulation de l'outil.

SCHEMA RECAPITULATIF

Pour récupérer un dossier depuis Github faire la commande git clone « url du dossier »
(ex git clone « https://github.com/AdelineMeister/ABCDEV_2306_MaD.git »)



--- FIN DU DOCUMENT ---