

On ouvre le terminal GitHub à l'aide de Git Bash

On vérifie la position du terminal dans l'ordi avec la commande pwd : - Pwd :
repérer l'emplacement du terminal dans le système

Le terminal dit qu'on se situe dans le dossier D-ploiment qui est dans le dossier vilne

- Cd : naviguer dans un dossier précis

On est allé dans le dossier Desktop donc sur le bureau de l'ordinateur

- Ls : montrer tous les fichiers/dossier/éléments du dossier

Le dossier repositories s'est bien créée dans Desktop

- Git clone : Sert à cloner son entrepôt :

Il s'agit d'un entrepôt privé, il faut donc rajouter le nom d'utilisateur dans le lien github
avant "github" et rajouter @ pour prouver qu'il s'agit bien du propriétaire du
repositories.

Le terminal demande donc de se connecter au compte pour prouver la connexion et
pouvoir cloner l'entrepôt

Le clonage est fait.

```
D-ploiment — -zsh — 80x56
Last login: Sun Oct  9 02:25:34 on ttys000
[vilne@MacBook-Air-de-chill ~ % git
usage : git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
          [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
          [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
          [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
          [--super-prefix=<path>] [--config-env=<name>=<envvar>]
          <command> [<args>]

Ci-dessous les commandes Git habituelles dans diverses situations :

démarrer une zone de travail (voir aussi : git help tutorial)
  clone      Cloner un dépôt dans un nouveau répertoire
  init       Créer un dépôt Git vide ou réinitialiser un existant

travailler sur la modification actuelle (voir aussi : git help revisions)
  add        Ajouter le contenu de fichiers dans l'index
  mv         Déplacer ou renommer un fichier, un répertoire, ou un lien symbolique
  restore    Restaurer les fichiers l'arbre de travail
  rm         Supprimer des fichiers de la copie de travail et de l'index

examiner l'historique et l'état (voir aussi : git help revisions)
  bisect     Trouver par recherche binaire la modification qui a introduit un bogue
  diff       Afficher les changements entre les validations, entre validation et
             copie de travail, etc
  grep       Afficher les lignes correspondant à un motif
  log        Afficher l'historique des validations
  show       Afficher différents types d'objets
  status     Afficher l'état de la copie de travail

agrandir, marquer et modifier votre historique
  branch     Lister, créer ou supprimer des branches
  commit     Enregistrer les modifications dans le dépôt
  merge      Fusionner deux ou plusieurs historiques de développement ensemble
  rebase     Réapplication des commits sur le sommet de l'autre base
  reset      Réinitialiser la HEAD courante à l'état spécifié
  switch     Basculer de branche
  tag        Créer, lister, supprimer ou vérifier un objet d'étiquette signé avec GPG

collaborer (voir aussi : git help workflows)
  fetch      Télécharger les objets et références depuis un autre dépôt
  pull       Rapatrier et intégrer un autre dépôt ou une branche locale
  push       Mettre à jour les références distantes ainsi que les objets associés

'git help -a' et 'git help -g' listent les sous-commandes disponibles et
quelques concepts. Voir 'git help <commande>' ou 'git help <concept>'
pour en lire plus à propos d'une commande spécifique ou d'un concept.
Voir 'git help git' pour un survol du système.
[vilne@MacBook-Air-de-chill ~ % git --version
git version 2.38.0
[vilne@MacBook-Air-de-chill ~ % pwd
/Users/vilne
```

```
D-ploiment — -zsh — 80x56
[vilne@MacBook-Air-de-chill ~ % pwd
/Users/vilne
[vilne@MacBook-Air-de-chill ~ % git clone
fatal : Vous devez spécifier un dépôt à cloner.

usage : git clone [<options>] [--] <dépôt> [<répertoire>]

    -v, --verbose           être plus verbeux
    -q, --quiet             être plus silencieux
    --progress             forcer l'affichage de l'état d'avancement
    --reject-shallow       ne pas cloner un dépôt superficiel
    -n, --no-checkout      ne pas créer d'extraction
    --bare                 créer un dépôt nu
    --mirror               créer un dépôt miroir (implique dépôt nu)
    -l, --local            pour cloner depuis un dépôt local
    --no-hardlinks         ne pas utiliser de liens durs locaux, toujours copier
    -s, --shared           régler comme dépôt partagé
    --recurse-submodules[=<spécificateur de chemin>]
                           initialiser les sous-modules dans le clone
    --recursive ...       alias pour --recurse-submodules
    -j, --jobs <n>        nombre de sous-modules clonés en parallèle
    --template <répertoire-modèle>
                           répertoire depuis lequel les modèles vont être utilisé
    --reference <dépôt>   dépôt de référence
    --reference-if-able <dépôt>
                           dépôt de référence
    --dissociate           utiliser seulement --reference pour cloner
    -o, --origin <nom>    utiliser <nom> au lieu de 'origin' pour suivre la bran
che amont
    -b, --branch <branche>
                           extraire <branche> au lieu de la HEAD du répertoire di
stant
    -u, --upload-pack <chemin>
                           chemin vers git-upload-pack sur le serveur distant
    --depth <profondeur>  créer un clone superficiel de cette profondeur
    --shallow-since <heure>
                           créer un clone superficiel depuis une date spécifique
    --shallow-exclude <révision>
                           approfondir l'historique d'un clone superficiel en exc
luant une révision
    --single-branch       cloner seulement une branche, HEAD ou --branch
    --no-tags             ne pas cloner les tags et indiquer aux récupérations f
utures de ne pas le faire
    --shallow-submodules  tous les sous-modules clonés seront superficiels
    --separate-git-dir <gitdir>
                           séparer le répertoire git de la copie de travail
    -c, --config <clé=valeur>
                           régler la configuration dans le nouveau dépôt
    --server-option <spécifique au serveur>
                           option à transmettre
    -4, --ipv4            n'utiliser que des adresses IPv4
    -6, --ipv6            n'utiliser que des adresses IPv6
    --filter <args>       filtrage d'objet
    --also-filter-submodules
                           appliquer les filtres de clone partiel aux sous-module
```

```

s
> --remote-submodules tous les sous-modules clonés utiliseront leur branche
re de suivi à distance
--sparse initialiser le fichier d'extraction clairsemée pour n'
e inclure que les fichiers à la racine
e --bundle-uri <uri> un URI pour télécharger des paquets avant de récupérer
o depuis le distant d'origine
i
i [vilne@MacBook-Air-de-chill ~ % git clone https://github.com/vilne92/D-ploiment.]
r git
Clonage dans 'D-ploiment'...
avertissement : Vous semblez avoir cloné un dépôt vide.
i [vilne@MacBook-Air-de-chill ~ % ls ]
Applications Documents Movies Public
D-ploiment Downloads Music Sites
Desktop Library Pictures redshift
[vilne@MacBook-Air-de-chill ~ % cd D-ploiment ]
[vilne@MacBook-Air-de-chill D-ploiment % git statusq ]
git : 'statusq' n'est pas une commande git. Voir 'git --help'.

La commande la plus ressemblante est
status
[vilne@MacBook-Air-de-chill D-ploiment % git status ]
Sur la branche main

Aucun commit

rien à valider (créez/copiez des fichiers et utilisez "git add" pour les suivre)
vilne@MacBook-Air-de-chill D-ploiment %

```