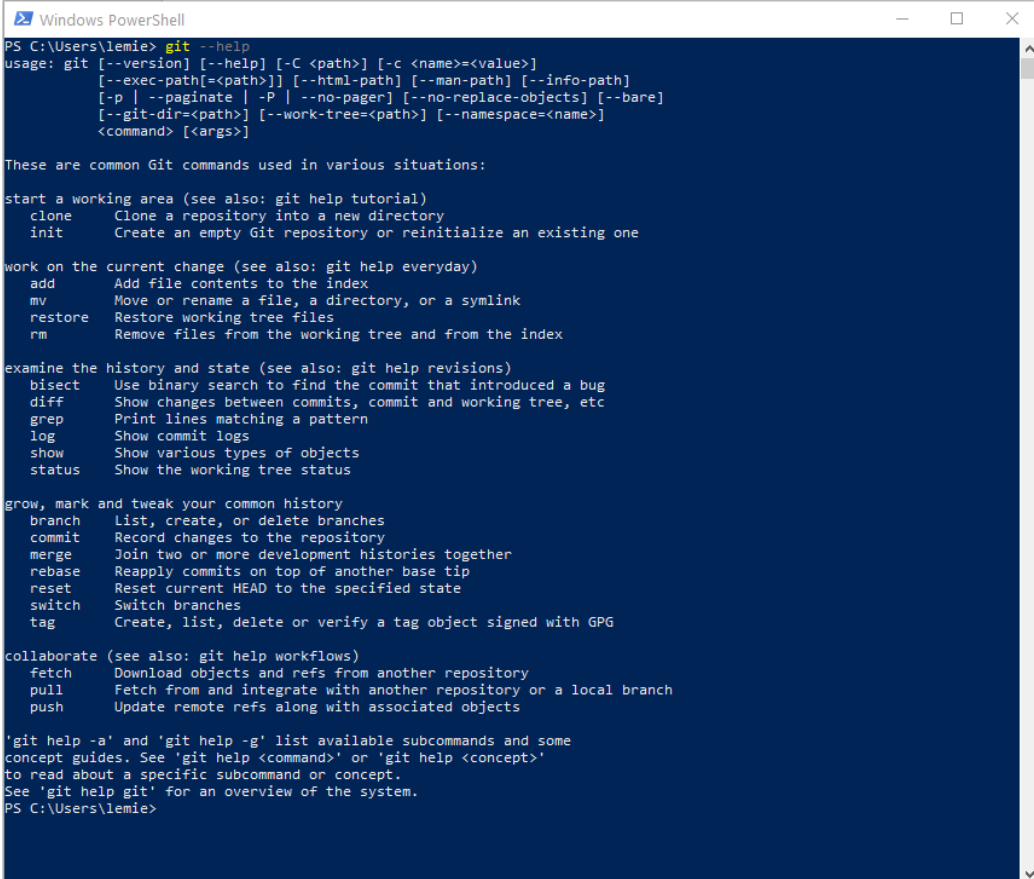


Labo 03 – Git & GitHub

Préparatifs

1. Installer **Git** sur son poste de travail

- Vous pouvez télécharger le logiciel **Git** à cette URL :
<https://git-scm.com/downloads>
- Au besoin, consultez les instructions pour l'installation :
<https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>
- Vérifiez que l'installation s'est bien déroulée et que le répertoire git a été ajouté à la variable d'environnement **Path** (C'est ce qui permet d'appeler le programme Git à l'aide de la commande '**git**', sans devoir spécifier le chemin complet du répertoire à chaque fois). Pour ce faire, dans un terminal, exécutez la commande '**git --help**'. Vous devriez voir quelque-chose qui ressemble à ça :



```
PS C:\Users\lemie> git --help
usage: git [--version] [--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>]
      <command> [<args>]

These are common Git commands used in various situations:


start a working area (see also: git help tutorial)
  clone      Clone a repository into a new directory
  init       Create an empty Git repository or reinitialize an existing one


work on the current change (see also: git help everyday)
  add        Add file contents to the index
  mv         Move or rename a file, a directory, or a symlink
  restore    Restore working tree files
  rm         Remove files from the working tree and from the index


examine the history and state (see also: git help revisions)
  bisect     Use binary search to find the commit that introduced a bug
  diff       Show changes between commits, commit and working tree, etc
  grep       Print lines matching a pattern
  log        Show commit logs
  show       Show various types of objects
  status     Show the working tree status


grow, mark and tweak your common history
  branch     List, create, or delete branches
  commit     Record changes to the repository
  merge      Join two or more development histories together
  rebase     Reapply commits on top of another base tip
  reset      Reset current HEAD to the specified state
  switch     Switch branches
  tag        Create, list, delete or verify a tag object signed with GPG


collaborate (see also: git help workflows)
  fetch      Download objects and refs from another repository
  pull       Fetch from and integrate with another repository or a local branch
  push       Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
See 'git help git' for an overview of the system.
PS C:\Users\lemie>
```

Si tout ne fonctionne pas comme prévu, contactez-moi et on tentera de régler ça ensemble, si possible dans les jours précédant le laboratoire!

2. Créer un compte **GitHub**

- Pour profiter pleinement des fonctionnalités de **Git**, nous aimerions avoir accès à un serveur en ligne. Cela va nous permettre d'avoir une copie de sécurité (backup) de notre code, de pouvoir partager notre code avec les utilisateurs, de travailler en collaboration avec d'autres programmeurs sur le même code, et bien plus encore! Nous utiliserons le service **GitHub**, bien connu, et gratuit pour l'utilisation que l'on compte en faire. Vous m'avez vu l'utiliser depuis plusieurs mois déjà! Allez à <https://github.com/> et créez-vous un compte. Choisissez bien le nom de votre profil, vous allez probablement l'utiliser pour plusieurs années à venir!
- Il peut être déroutant de se repérer sur l'interface web de GitHub, mais à force d'utilisation, vous finirez par vous y retrouver. Nous explorerons le site ensemble pendant le laboratoire et vous apprendrez à créer et gérer des répertoires (aka *Repository*). Pour ceux que ça intéresse, vous pouvez aussi utiliser une interface en ligne de commande (aka CLI) qui communique directement avec l'API de GitHub (<https://cli.github.com/>).

3. Avoir une vue d'ensemble de **Git** et ses fonctionnalités

- Pendant la séance du cours précédant le laboratoire, Alix devrait vous présenter **Git**, vous expliquer à quoi ça sert et vous montrer les fonctionnalités les plus utiles. Néanmoins, **Git** est à première vue assez complexe et il vous serait bien profitable d'aller consulter quelques ressources supplémentaires. YouTube est un bon endroit pour ça; j'ai particulièrement aimé cette vidéo de Brian Yu : <https://www.youtube.com/watch?v=eulnSXkhE7I>. C'est une vidéo de 45 minutes (la dernière partie concerne un autre sujet) mais c'est du temps bien investi!
- Il existe aussi des aide-mémoire (aka Cheat Sheet) qui sont bien utiles à avoir sous la main. Vous pouvez télécharger celle-ci : <https://i.redd.it/8341g68g1v7y.png>

Vous êtes maintenant prêt à faire les exercices que j'ai préparé pour vous! On va les faire ensemble à la prochaine séance de laboratoire.