

Synthèse de la journée Console & Git

SP5

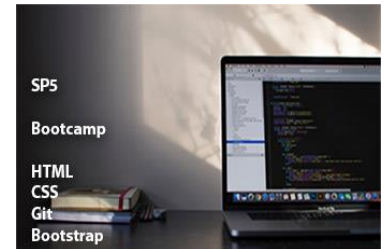


Table des matières

Synthèse de la journée Console & Git	1
Console.....	2
Shell – « coque »	2
GUI - Graphical User Interface	2
CLI - Command Line Interface	2
Bash.....	3
Exemple de console	3
Commandes bash – Principales :	3
Les commandes git et Github	5
Gestionnaire de versions	5
Github	5
Github desktop.....	6
Netlify.....	7
Forker un Repo.....	7
Source pour aller plus loin	7
Récap d'une ancienne apprenante	7
Pour prendre de l'avance sur le html&css	7

Console

La console l'outil permettant d'avoir une interface textuelle du système d'exploitation du système.

- Windows
- Linux ou Mac

Shell – « coque »

Un shell ("coque" en français) est une interface système. Le shell nous permet d'accéder à notre système d'exploitation (OS) et d'interagir avec lui. En termes simples, le shell est un programme qui reçoit des commandes qu'on va écrire depuis le clavier et qui les passe au système d'exploitation afin qu'elles soient exécutées.

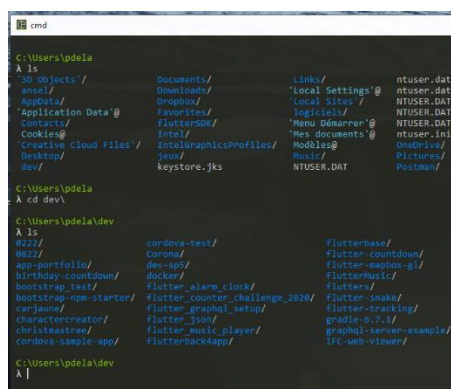
Le terme shell provient d'une analogie : celle du noyau et de la coque de noix. L'idée ici est qu'un OS est composé de deux parties : le shell (la coque) et le kernel (le coeur). Pour accéder au coeur de la noix, c'est-à-dire à la partie comestible, il faut passer par la coque. De même, pour envoyer nos commandes au coeur de notre OS afin qu'il les exécute nous devons passer par le shell.

Un shell, c'est-à-dire à proprement parler la coque logicielle d'un système d'exploitation peut se présenter sous deux formes : soit une interface en ligne de commande, soit une interface graphique utilisateur. Notez cependant que la plupart des gens utilisent le terme "shell" pour désigner l'interface en ligne de commande uniquement.

GUI - Graphical User Interface

L'interface graphique utilisateur ou GUI pour "Graphical User Interface") est ce que vous voyez lorsque vous utilisez votre ordinateur de manière "classique". Elle est composée d'un écran d'accueil, d'un bureau, d'icônes pour représenter les fichiers et les dossiers et de champs de recherche, de filtres, etc.

CLI - Command Line Interface



```

C:\Users\pdela> ls
 3D Objects / Documents / Links / ntuser.dat
 ansel / Downloads / 'Local Settings' / ntuser.dat
 AppData / Dropbox / 'Local Sites' / NTUSER.DAT
 Application Data / Favorites / hgt-ids / NTUSER.DAT
 Contacts / flutterSDK / 'Menu Démarrer' / ntuser.dat
 Cookies / Intel / 'Mes documents' / ntuser.ini
 Creative Cloud Files / IntelGraphicsProfiles / Modis / OneDrive /
 Desktop / Java / NTUSER.DAT / Pictures /
 dev / keystore.jks / Postman /

C:\Users\pdela> cd dev\

C:\Users\pdela\dev> ls
 0222 / cordova-test / flutterbase /
 022 / corona / flutter-countdown /
 app-portfolio / dev-app / flutter-mapbox-gl /
 birthday-countdown / docker / flutter-music /
 bootstrap-test / flutter_alarm_clock / flutter /
 bootstrap-npm-starter / flutter_counter_challenge_2020 / flutter-snake /
 carlaine / flutter_gradml_setup / flutter-tracking /
 charactercreator / flutter_ios / gradle-0.7.1 /
 christmas / flutter_music_player / graphql-server-example /
 cordova-sample-app / flutterbackapp / ifc-web-viewer /

C:\Users\pdela\dev>
  
```

Une interface en ligne de commande ou CLI pour "Command Line Interface" est beaucoup plus austère qu'une GUI et peut paraître primitive par rapport à une interface graphique.

Celle-ci se présente sous la forme d'une **grande boîte noire** dans laquelle l'utilisateur va taper des commandes (généralement une commande par ligne) pour interagir avec son OS.

Dans une CLI, il n'y a aucun usage de la souris : tout se fait via l'entrée de commandes tapées au clavier. Nous allons voir ensemble durant ce cours que les CLI sont en fait très

puissantes et vont nous permettre de réaliser des opérations **complexes beaucoup plus rapidement** qu'avec une GUI.

Les shells CLI exigent donc que l'utilisateur **connaisse certaines commandes** et leur syntaxe et comprenne les concepts relatifs au langage de script spécifique au shell utilisé (comme Bash pour le shell de Mac OS par exemple).

Bash

Le shell est la coque logicielle d'un système d'exploitation. Nous allons utiliser le shell pour envoyer des instructions à notre ordinateur.

Lorsqu'on utilise une interface en ligne de commandes, nous allons devoir envoyer nos commandes sous un certain format compréhensible par notre OS. Ce "format" va en fait être un langage de programmation spécifique au shell utilisé. On parle aussi par simplicité de "type de shells".

Il existe de nombreux shell Unix différents. Le plus utilisé est **Bash** ("Bourne Again SHell"). Bash est notamment le shell utilisé par défaut par **les systèmes OS X** (Mac, vrai jusqu'à la mise à jour Catalina prévue fin 2019) et il sera notre langage de commande de référence pour ce cours.

L'équivalent de Bash pour Windows est PowerShell. Ce cours est centré autour de Bash et nous ne parlerons donc pas de PowerShell. La plupart des commandes de base qu'on va pouvoir taper dans le Terminal (Mac) ou dans l'invite de commande (Windows) possèdent cependant une syntaxe similaire.

Définition console : <https://doc.ubuntu-fr.org/console>

Exemple de console

<https://cmder.net/>

<https://hyper.is/>

Commandes bash – Principales :

Echo « hello » -> affiche hello

« pwd » print working directory : voir où on est

"ls" → pour lister les fichiers

"cd" → command directory : pour se déplacer

"mkdir [le_nom_du_dossier]" → pour créer un dossier

"rm -rf [le_nom_du_dossier]" → pour supprimer un dossier

"mv source destination" → pour renommer un dossier

"touch [le_nom_du_fichier]" → pour créer un fichier

"echo "test" > index.html" -> insérer le mot test dans le fichier index.html

<https://www.shellunix.com/commandes.html>

Pour aller plus loin, d'autres commandes intéressantes :

Commandes Shell :

"cat nom_du_fichier" pour voir le contenu d'un fichier

"vi nom_du_fichier" pour éditer le fichier

Trouver un fichier : *find . -name 'nom_du_fichier'*

Se connecter à distance : ssh user@host

Pour télécharger les ressources sur internet : wget [adresse_url]

tache shedulé (programmé) = cron

synchronisation de dossier = rsync

se connecter à mysql en ligne de commande = mysql

compresser un dossier : zip [nom_du_dossier]

Décompresser : unzip [nom_du_dossier]

Pour avoir l'aider sur les commande : [nom_de_la_commande] -help

voir dans un fichier : cat [nom_du_fichier] ou less [nom_du_fichier]

<https://www.shellunix.com/commandes.html>

https://developer.mozilla.org/fr/docs/Learn/Tools_and_testing/Understanding_client-side_tools/Command_line

Les commandes git et Github

Git et GitHub sont deux choses différentes :

- Git est un **gestionnaire de versions**. Vous l'utiliserez pour créer un dépôt local et gérer les versions de vos fichiers.
- GitHub est **un service en ligne qui va héberger votre dépôt**. Dans ce cas, on parle de dépôt distant puisqu'il n'est pas stocké sur votre machine.

Gestionnaire de versions

Le gestionnaire de versions permet de **garder en mémoire** :

- chaque modification de chaque fichier ;
 - pourquoi elle a eu lieu ;
 - et par qui !
- **Si vous travaillez seul**, vous pourrez garder l'historique de vos modifications ou revenir à une version précédente facilement.
 - **Si vous travaillez en équipe**, plus besoin de mener votre enquête ! Le gestionnaire de versions fusionne les modifications des personnes qui travaillent simultanément sur un même fichier. Grâce à ça, vous ne risquez plus de voir votre travail supprimé par erreur !

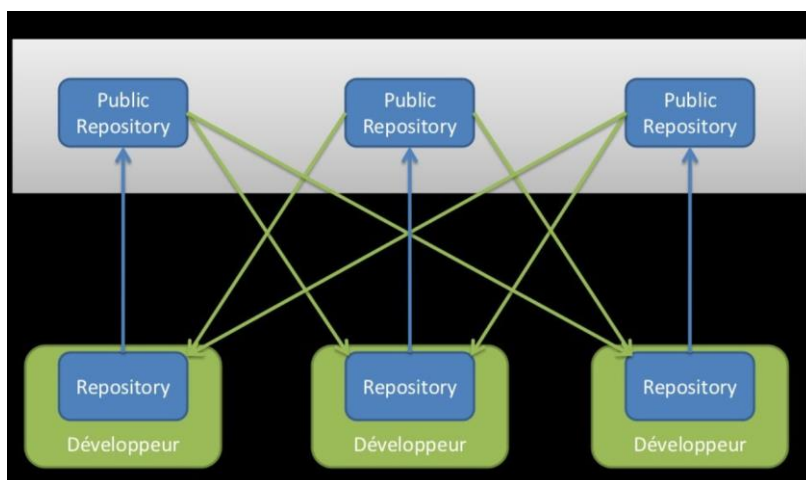
<https://openclassrooms.com/fr/courses/7162856-gerez-du-code-avec-git-et-github/7165703-decouvrez-la-magie-du-controle-de-versions>

Github

A un haut niveau, GitHub est un site web et un service de cloud qui aide les développeurs à stocker et à gérer leur code, ainsi qu'à suivre et contrôler les modifications qui lui sont apportées.

<https://kinsta.com/fr/base-de-connaissances/base-de-connaissances-github/>

https://youtu.be/gp_kOUVOYMw (LES BASES DE GIT tuto débutant – 17min)



GitHub, des chiffres qui donnent le vertige

Le site est né en 2007. Il atteint aujourd'hui des sommets :

- 40 millions de développeurs l'utilisent dans le monde.
- Près de 100 millions de projets hébergés.
- 2,9 millions d'organisations travaillent sur GitHub.
- Il a été racheté par Microsoft environ 7,5 milliards de dollars. 💰

Prix

Pour les projets Open Source, GitHub est gratuit 😊 C'est cette particularité qui lui a permis de se propulser dans le haut du classement des hébergements de projet.

GitHub gagne de l'argent grâce à l'hébergement des projets privés. Il en existe beaucoup avec abonnement et cela permet aux autres projets de pouvoir être partagés : un cercle vertueux.

Pourquoi

GitHub apporte deux éléments essentiels pour les développeurs : un accès au Git et un contrôle des versions de leur code.

Git

Rappel, Git est un gestionnaire de versions. Vous l'utiliserez pour créer un dépôt local et gérer les versions de vos fichiers.

Cet outil a été développé par le père de Linux, **Linus Torvalds**.

Télécharger : <https://git-scm.com/downloads>

Principales commandes git

```
$ git init / git clone
$ git add
$ git commit
$ git checkout
$ git merge
$ git push
$ git pull = fetch + merge
```

<https://git-scm.com/docs>

Github desktop

Le logiciel permettant de se passer des lignes de commandes pour utiliser github 😊

Tuto : <https://youtu.be/h9YZRKhfXv8> (Utiliser Github Desktop pour notre projet – 12min)

<https://desktop.github.com/>

Netlify

Netlify est un des leaders de l'hébergement de sites statiques. Fonctionne avec Github.

Tuto : <https://youtu.be/YzrlcOLOJtw> (Déployer un site web avec Netlify – 7min)

<https://www.netlify.com/>

Forker un Repo

Un fork est une copie d'un dépôt. Forker un dépôt vous permet d'expérimenter librement des modifications sans toucher au projet original.

<https://www.christopheducamp.com/2013/12/16/forker-un-repo-github/>

Ignorer des fichier avec .gitignore

le fichier .gitignore permet de liste les fichiers ou dossier à ne pas versionner. C'est pratique !

Vider les caches de git

Util par exemple lorsqu'on ajoute un fichier .gitignore

When you think your git is messed up, you can use this command to do everything up-to-date.

```
git rm -r --cached .
git add .
git commit -am 'git cache cleared'
git push
```

Git / comit visualisation

<https://youtu.be/vsMUTsFdZr4>

<https://youtu.be/r0ji8FDNTj0>

Github copilote

<https://copilot.github.com/>

<https://youtu.be/CuO5X9CupUE>

Source pour aller plus loin

<https://fr.slideshare.net/julienblin2/prsentation-de-git>

<https://openclassrooms.com/fr/courses/1233741-gerez-vos-codes-source-avec-git>

Récap d'une ancienne apprenante

<https://bootcampsp5.netlify.app/>

Pour prendre de l'avance sur le html&css

<https://openclassrooms.com/fr/courses/1603881-apprenez-a-creez-votre-site-web-avec-html5-et-css3/1604361-creez-votre-premiere-page-web-en-html>

A vous de coder !