CHRISTOPHE BOUTHIER - TELECOM NANCY

COLD #1

COMMANDES ET OUTILS LINUX POUR LE DEVELOPPEUR

OUTILS

Se débrouiller avec la ligne de commande :

naviguer dans le système de fichier -

gérer les droits -

gérer les processus -

trouver de l'information -

Git

Installer un logiciel ou une librairie

les gestionnaires de paquets -

Les expressions régulières

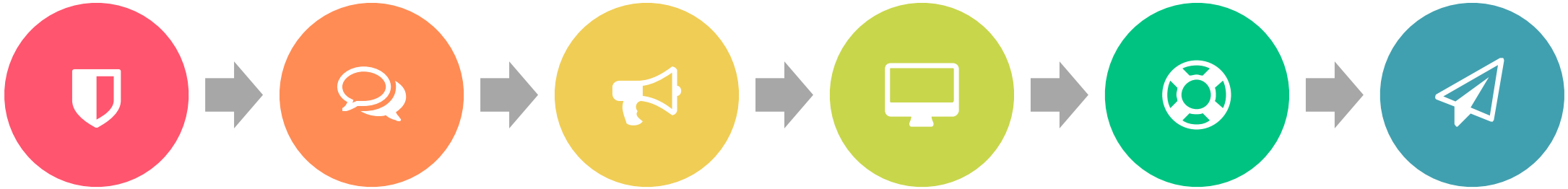
Outils rapides en python

DEROULEMENT DES TDS

QCM
Pour tester ce que vous avez appris lors du TD/CM précédent

EXPLICATIONS
Détails du ou des points importants de ce TD

CORRECTIONS
Solutions et corrections des exercices dirigés

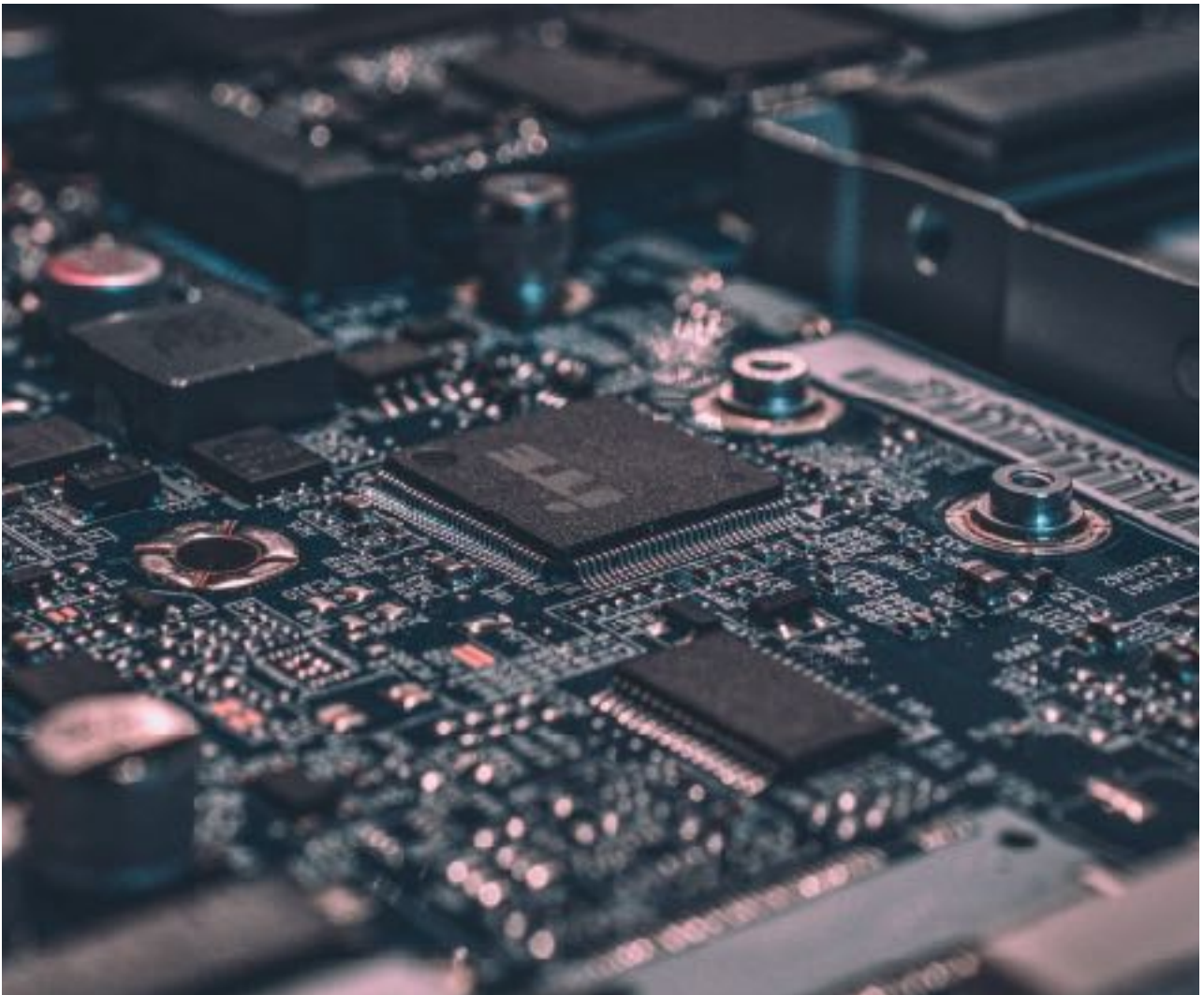


QUESTIONS
Réponses aux questions que vous avez pu avoir en faisant les exercices en autonomie

EXERCICES
Réalisation des exercices dirigés sur machine

AUTONOMIE
Exerices à faire chez soi en autonomie pour solidifier les acquis et préparer le TD suivant

ARCHITECTURE D'UN ORDINATEUR



- **CPU**
  Execute les instructions
- **RAM**
  Mémoire d'exécution
- **DISQUE**
  Stockage statique
- **PERIPHERIQUES**
  Son, réseau
- **ENTREE UTILISATEUR**
  Clavier, souris
- **SORTIE UTILISATEUR**
  Carte graphique, ecran

→ TACHES

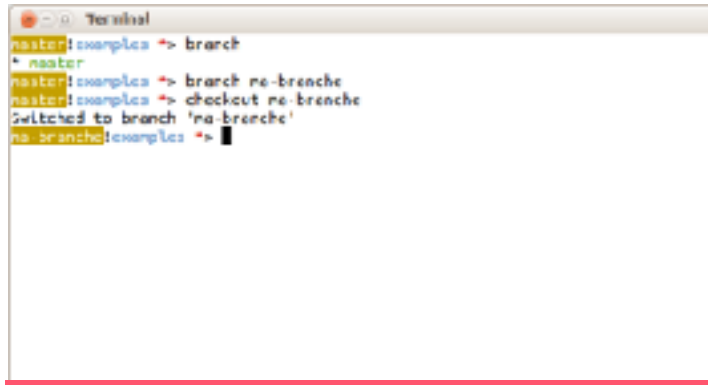
SYSTEME D'EXPLOITATION

- Couche d'abstraction hardware
  - Drivers
- Gère :
  - Processus
  - Ressources (mémoire, stockage, CPU)
- Fonctionnalités communes applications
- Interface Utilisateur
  - Gestion des fichiers
  - Lancement des taches

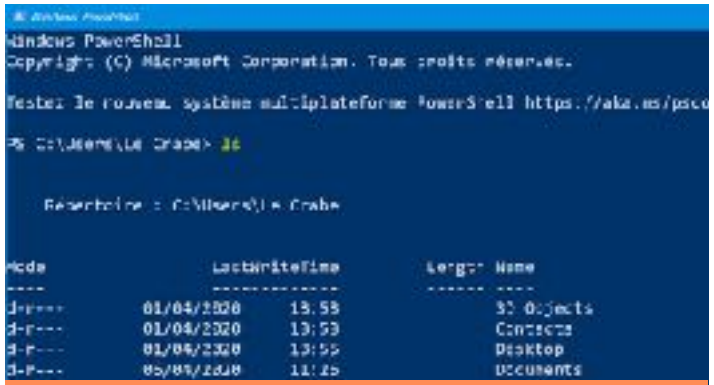
LIGNE DE COMMANDE

- Shell
  - « coquille » : encapsule l'OS
  - manipulation de l'OS
- Interface « au plus proche » de l'OS
  - Donne des ordres directement
  - Grande flexibilité (options)
  - Manipule les objets directement
- Automatisable
- ATTENTION !
  - Moins de garde-fou…

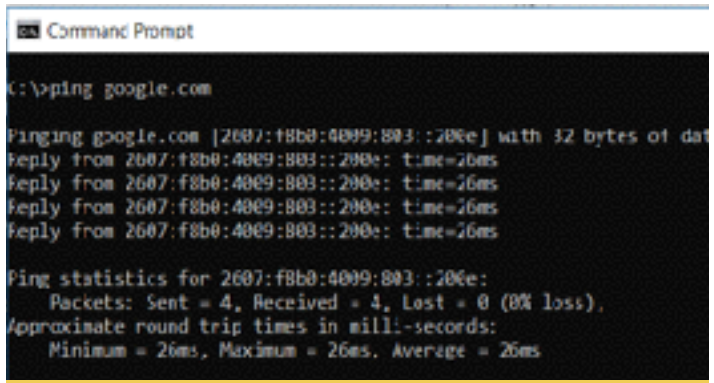
DIFFERENTS SHELL



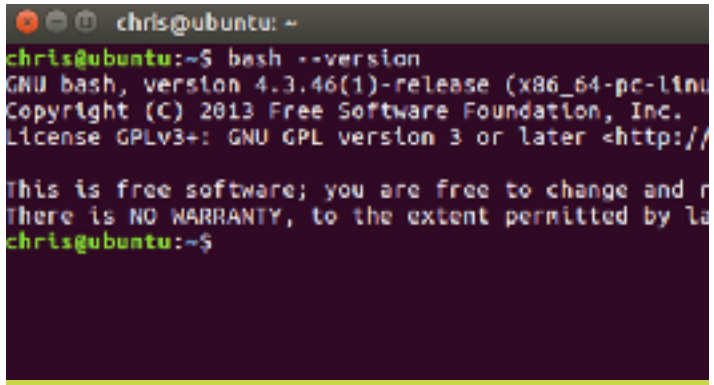
Terminal



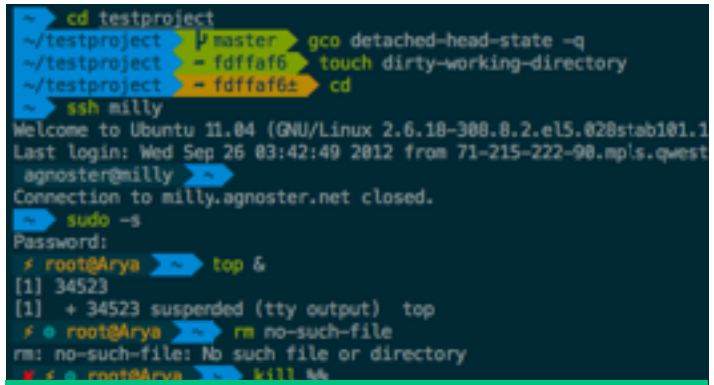
PowerShell



CMD



Bash



Zsh

Fish

POURQUOI LA LIGNE DE COMMANDE ?

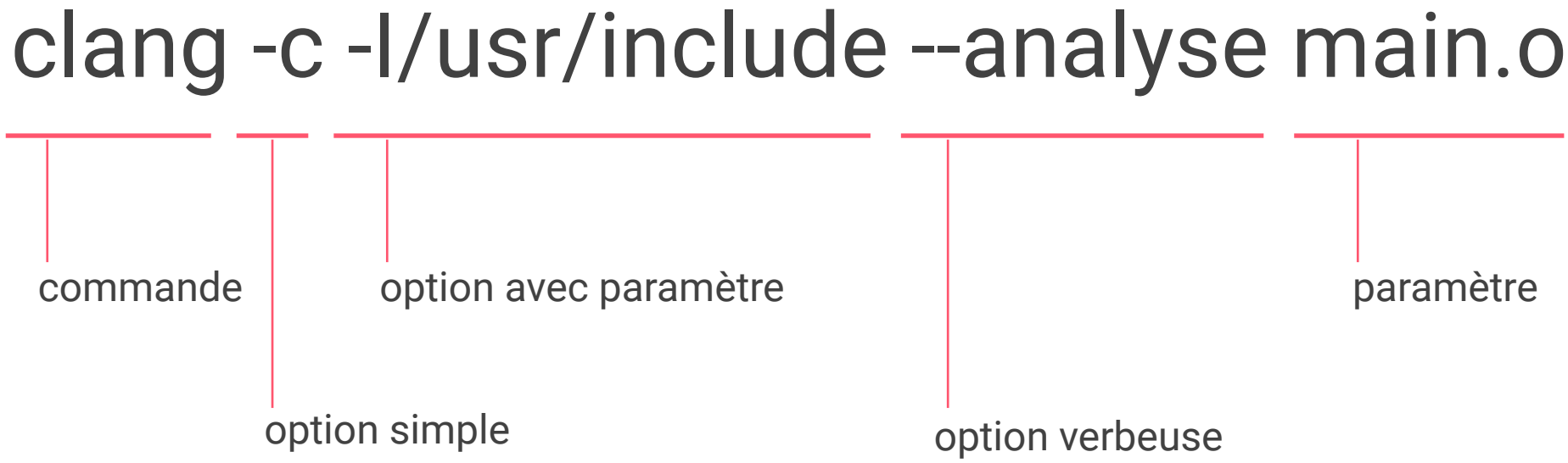
- « L'interface graphique est plus pratique, rien à se rappeler, pourquoi ne pas l'utiliser ? »
  - Tout n'est pas faisable en GUI
  - Tout n'est pas facile en GUI (fichiers multiples…)
  - L'interface graphique n'est pas automatisable
- GUI = fait pour les utilisateurs
- CLI = fait pour les « Power Users »
  - Ingénieur == Power User !

ETRE CONFORTABLE AVEC LA CLI

- Ne pas avoir peur de l’utiliser
- Savoir passer de la GUI à la CLI et inversement
- Connaitre les commandes de bases
- Connaitre les principes de bases
- Connaitre les outils de bases

- Avoir autant de facilité à l’utiliser que la GUI

ARCHITECTURE D'UNE COMMANDE



TIPS

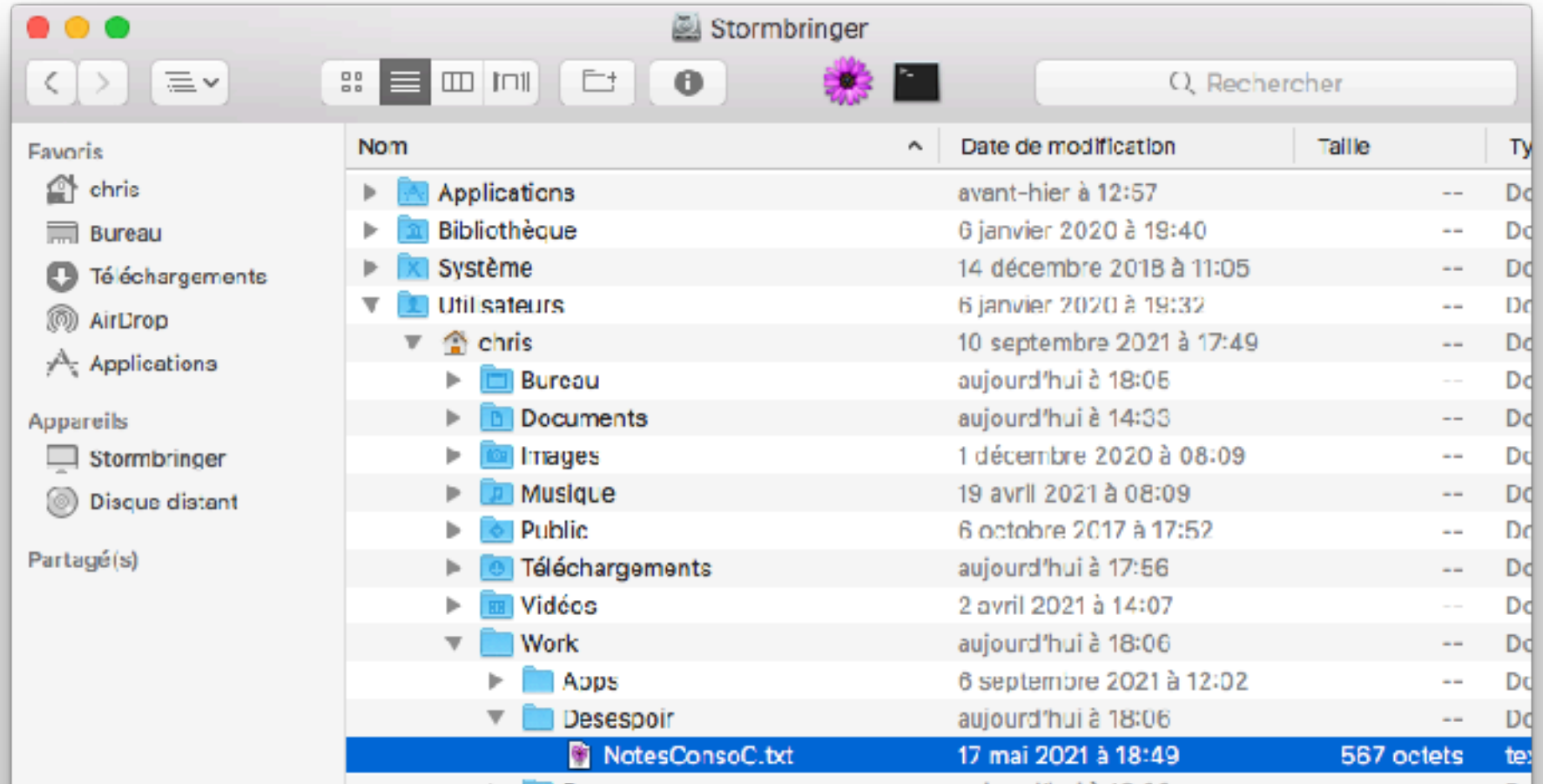
- Commutativité des options :
  - grep -i -r == grep -r -i
  - Attention si l'option a un paramètre !
    - find -name toto -i == find -i -name toto
- Combinaison des options
  - grep -i -r == grep -ir
- Touche Tab : auto-completion
  - Evite de se fatiguer
  - Evite les erreurs
  - Aide mémoire

SYSTEME DE FICHIERS

- Stockage
- Arborescence de répertoires et de fichiers
  - Manipulée par l'interface graphique
- Notion de CHEMIN
  - Comme URL
  - « Adresse » du fichier sous forme de texte
  - Suite des répertoires contenant le fichier
  - Séparateur : « / »

ANATOMIE D'UN CHEMIN

/Utilisateurs/chris/Work/Desespoir/NotesConsoC.txt



2 TYPES DE CHEMINS

- Absolu :
  - Commence par « / »
  - Part de la racine du disque
  - /Users/chris/Work
  - /var/www/html

- Relatif :
  - Ne commence PAS par « / »
  - Part de « là ou on est »
  - Work
  - chris/Work

« LA OU ON EST »

- Répertoire « courant »
  - Là ou on lance la commande
  - Fait partie de l'environnement
  - Toute tache a un répertoire courant
    - Même si lancé depuis GUI
- Répertoire de départ :
  - Home directory
  - Début de « l'espace personnel »
  - Raccourci : « ~ »

QUELQUES COMMANDES

- cd :
  - Change Directory
  - Permet de se déplacer dans le système de fichier
  - Prend en paramètre un chemin (absolu ou relatif)
- pwd :
  - Print Working Directory
  - Affiche le répertoire courant
- ls :
  - LiSt
  - Affiche le contenu du répertoire courant

EXEMPLE

- cd ~
- pwd
  - /Users/chris
- ls
  - Work
- cd Work
- pwd
  - /Users/chris/Work
- ls
  - Notes.txt

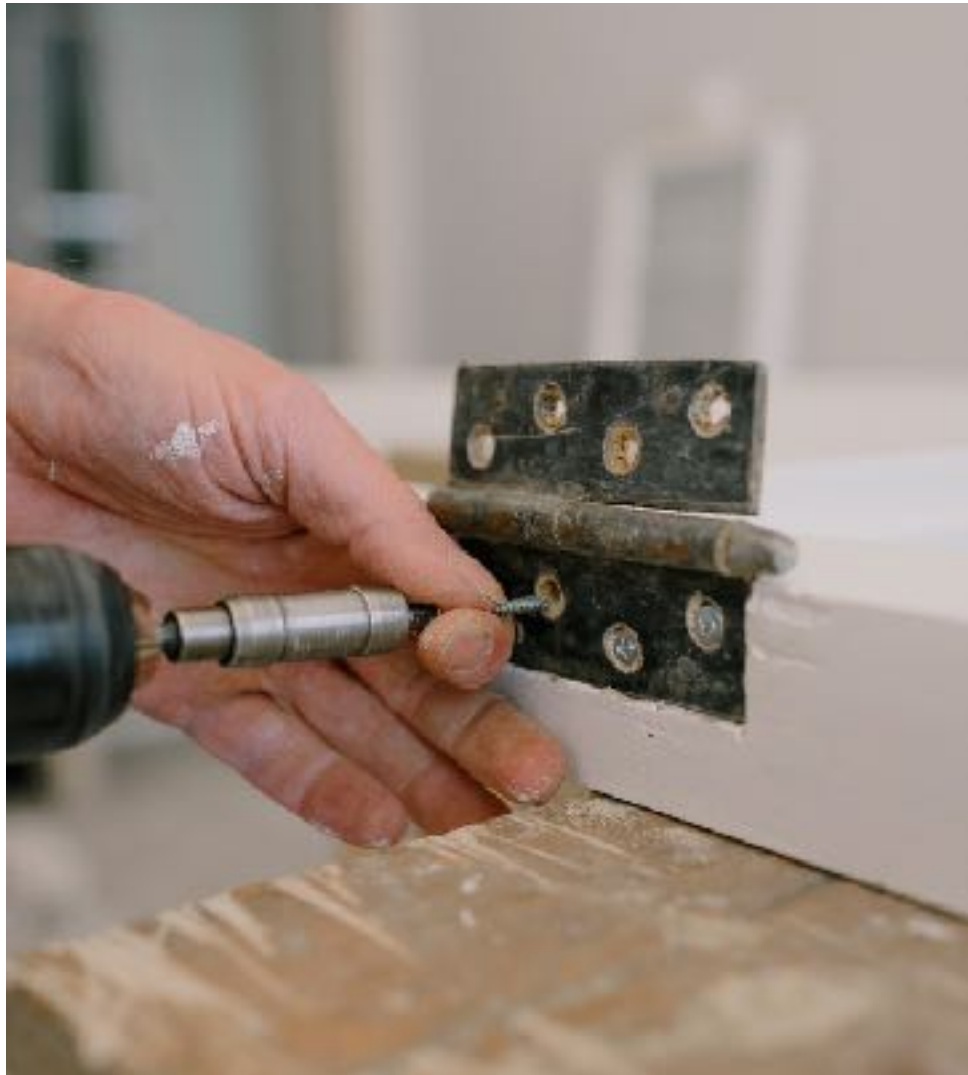
- cd /Users/chris/Work
- pwd
  - /Users/chris/Work
- ls
  - Notes.txt

TRAVAILLER, TRAVAILLER…

- Les commandes : pendant les TDs
  - C’est en faisant qu’on apprend
  - Il faut les utiliser pour les connaitre
- Utiliser le plus souvent possible la CLI
  - Préférer la CLI plutôt que la GUI
- Se renseigner sur les commandes
  - Commande « man » pour avoir la doc
  - https://explainshell.com
    - Permet d’expliquer une commande

Installation

- Besoin d’un logiciel ou d’une librairie
  - Comment le trouver ?
  - Ou l’installer ?
  - Comment gérer les dépendances ?
  - Comment gérer les mises à jour ?
- Au niveau de l’OS
- Au niveau d’un langage
  - Python, Java, …
  - (pas C)



GESTIONNAIRE DE PAQUETS

- Recherche du logiciel / de la librairie
- Récupère la liste des dépendances
  - Avec les bons numéros de versions
- Télécharge les dépendances
- Installe les dépendances
- Télécharge le binaire
- Installe le binaire et les fichiers liés
- Permet la mise à jour en cas de nouvelles versions
- Gère (parfois) les dépendances obsolètes

EXEMPLES

LINUX
apt

PYTHON
pip

JAVA
graddle

NODEJS
npm

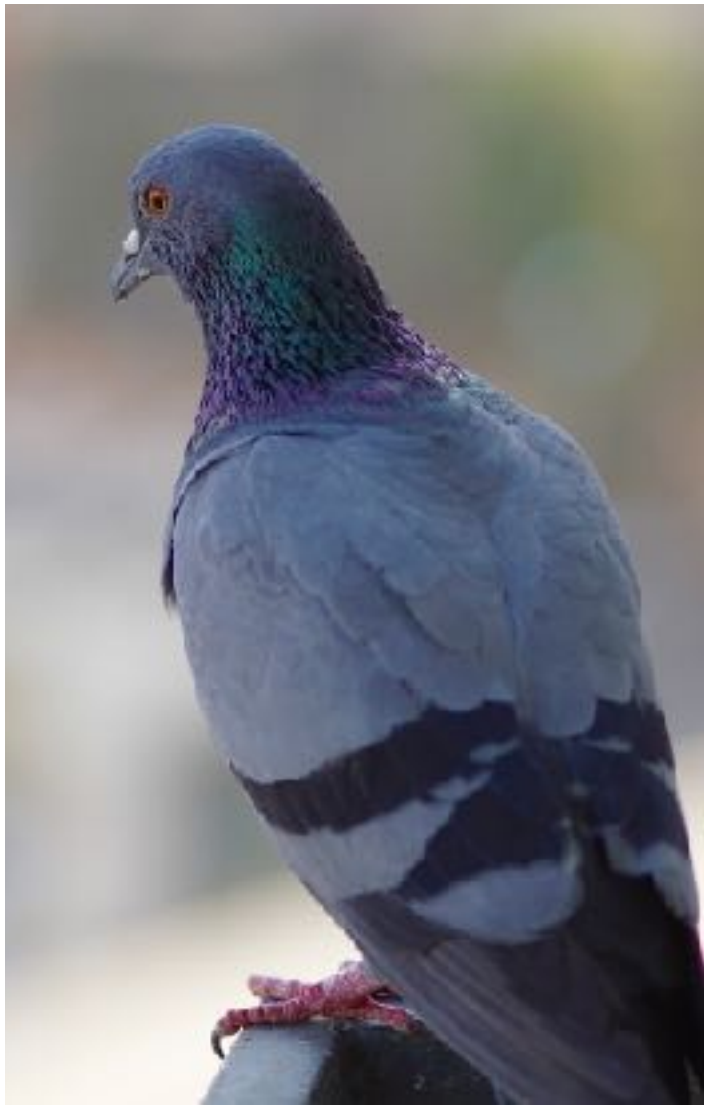
OSX
homebrew

Versions

- Comment garder les anciennes versions de ses fichiers ?
  - code_v1.c
  - code_v2.c
  - code_final.c
  - code_very_final.c
  - code_vraiment_vraiment_final.c
  - code_correction.c
  - code_correction_final.c
  - …

Partage

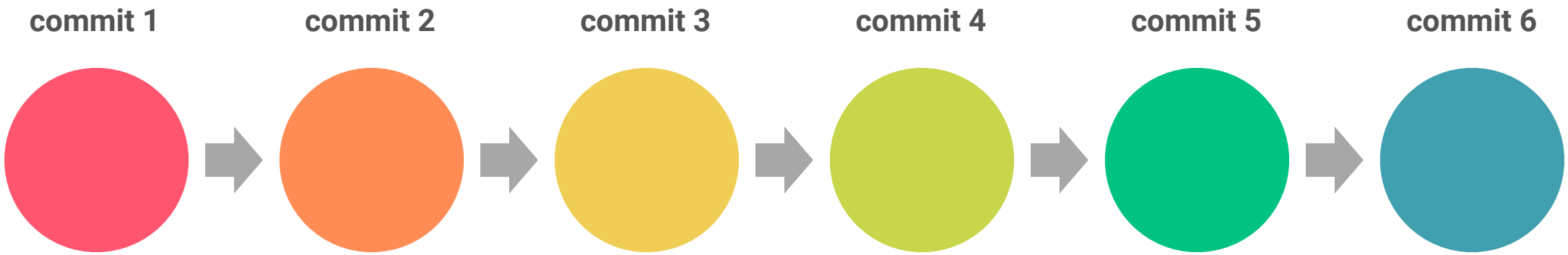
- Comment partager mon code avec d’autres ?
  - Clef USB
  - Mail
  - Facebook
  - Google Drive
  - Pigeon Voyageur
  - …
- Comment partager les nouvelles versions ?



GIT

- Gestion de version
  - Gère les répertoires et les fichiers
  - Permet les retours en arrière
- Partage
  - Dépose des fichiers sur un serveur
  - Facilité pour récupérer les fichiers
  - Facilité pour récupérer les modifications
- Autres fonctionnalités
  - Evolution de code en parallèle

ARBRE DE VERSIONS



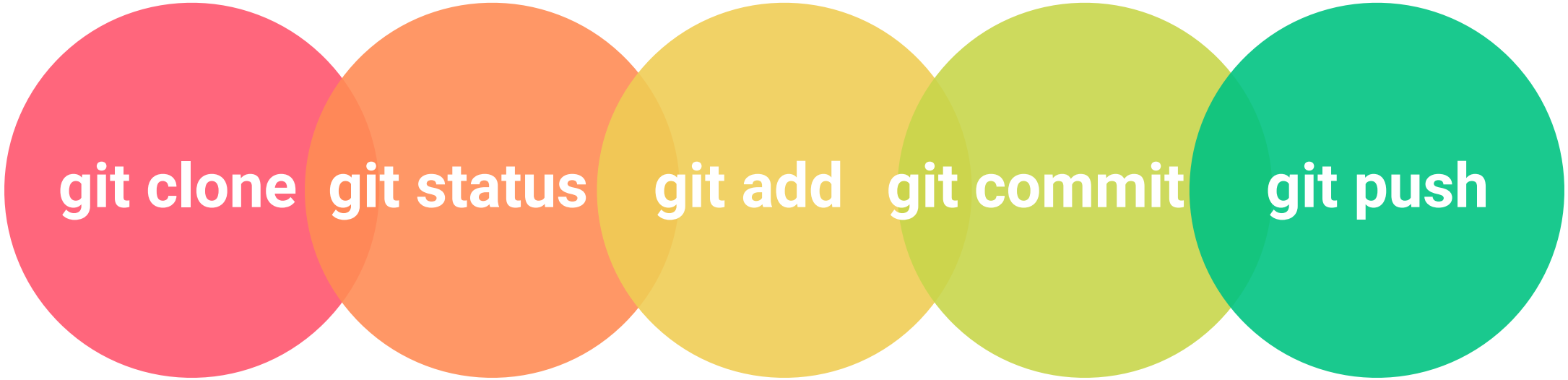
Relation causale

Le commit 2 dépend des modifications apportées par le commit 1

Dernière version

Version récupérée lorsqu'on récupère les fichiers du serveur

WORKFLOW DE BASE



git clone
Récupère le projet depuis le serveur

git status
Vérifie l'état des modifications

git add
Prend en compte les modifications

git commit
Sauvegarde une nouvelle version avec ces modifications

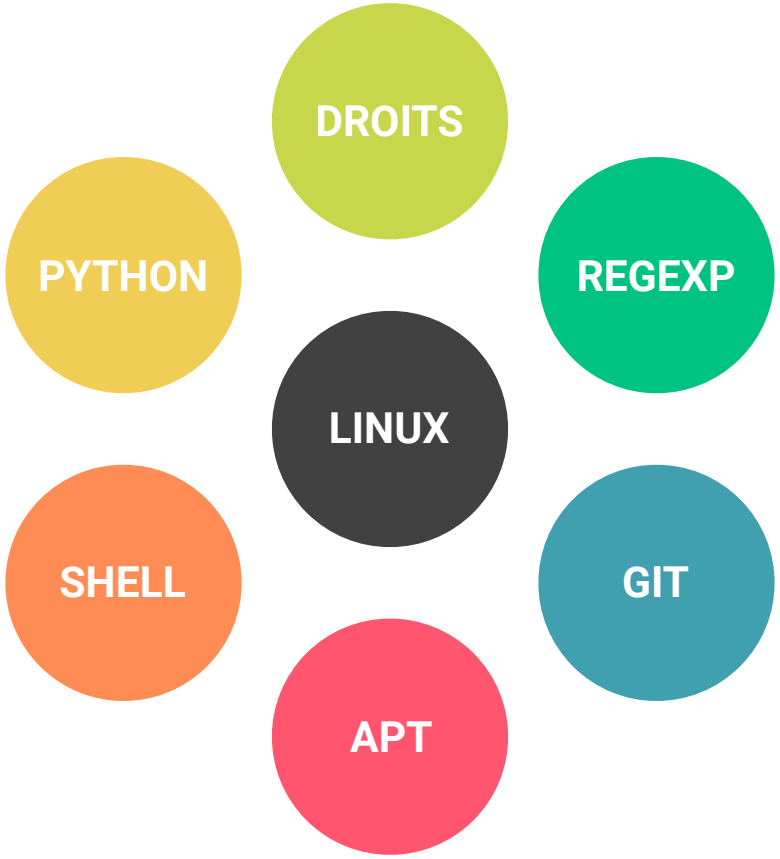
git push
Envoi la ou les nouvelles versions sur le serveur

SAVOIR FAIRE

TRAVAILLER DANS LES TDS
Faire les exos sans attendre la correction

POSER DES QUESTIONS
Il n'y a pas de questions bêtes

FAIRE LES EXOS EN AUTONOMIE
Ne pas se dire que l'on fera tout à la fin



PRIVILEGIER LA CLI
Au lieu de toujours utiliser l'interface graphique

PRATIQUER UN PEU CHAQUE JOUR
Plutôt qu'intensément un seul jour par semaine

ACQUERIR UN SAVOIR-FAIRE
Et non pas chercher à juste avoir une bonne note