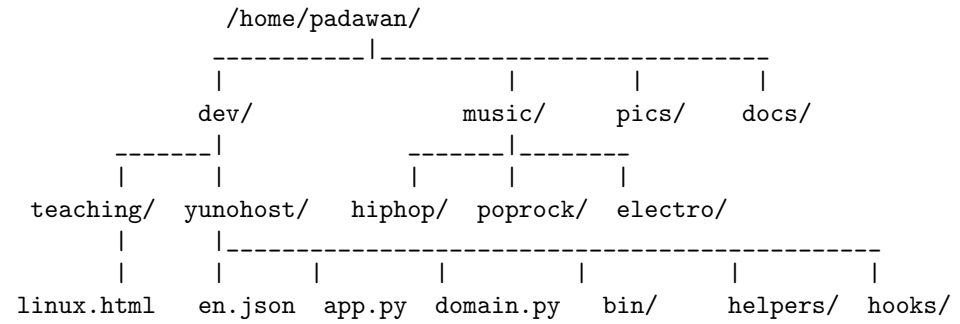
RAPPELS DES COMMANDES UNIX

On suppose qu'il existe l'arborescence suivante dans votre répertoire personnel (`home`) :



Vous vous trouvez dans le répertoire `music/electro/` (on nomme ce répertoire le *répertoire courant*). Depuis ce répertoire, tous les chemins suivants sont équivalents :
(absolu) `/home/padawan/dev/yunohost/app.py`
(absolu) `$HOME/dev/yunohost/app.py`
(absolu) `~/dev/yunohost/app.py`
(relatif) `../../dev/yunohost/app.py`
(relatif) `../../pics/./../dev/teaching/../yunohost/app.py`
et aussi : `/var/cache/../../home/./padawan/dev/yunohost/app.py`

Commandes indispensables

`man`	`commande`	(**man**ual) Obtenir des informations sur une commande
`passwd`		(**passw**or**d**) Changer son mot de passe
`cd`	`destination`	(**c**hange **d**irectory) Changer de répertoire (si pas de destination, va dans `$HOME`)
`ls`	`repertoire`	(**ls**t) Lister le contenu du répertoire
`-l`		Ajoute des détails (permissions, date de modification, taille)
`-a`		Liste aussi les fichiers cachés (commençant par .)
`-t`		Liste en triant suivant la date de modification
`-h`		Affiche les tailles de fichiers en un format lisible facilement
`mkdir`	`repertoire`	(**m**a**k**e **dir**ectory) Créer un répertoire
`mv`	`source(s) destination`	(**m**o**v**e) Renommer un fichier (ou déplacer plusieurs fichiers dans un répertoire)
`cp`	`fichier destination`	(**c**o**p**y) Copier un fichier
`-r`	`dossier destination`	Copier récursivement un dossier
	`fichier(s) repertoire`	Copie des fichiers dans un répertoire déjà existant
`rm`	`fichier(s)`	(**r**e**m**ove) Supprimer **définitivement** des fichiers
`-r`		Suppression récursive (et **définitive**) de répertoires
`-i`		Demande confirmation avant les suppressions

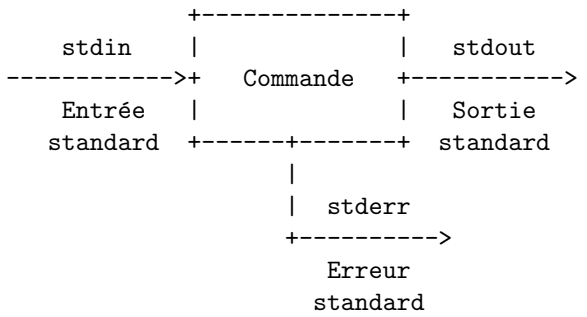
Commandes de base

`echo`	`chaîne`	Afficher une chaîne de caractère dans la sortie
`touch`	`fichier`	Créer un fichier (ou changer sa date de modification sans rien changer)
`pwd`		(**p**rint **w**orking **d**irectory) Obtenir le nom du répertoire courant
`whoami`		(**who am I**) Obtenir votre nom d'utilisateur
`date`		(**date**) Afficher la date
`chmod`	`permissions fichier`	(**ch**ange **mod**e) Changer les permissions d'un fichier
`chown`	`utilisateur fichier`	(**ch**ange **own**er) Changer le propriétaire d'un fichier
`chgrp`	`groupe fichier`	(**ch**ange **gr**ou**p**e) Changer le groupe d'un fichier
`du`	`fichier`	(**d**isk **u**sage) Connaître l'espace disque utilisé par un fichier
`-h`		Donne les tailles dans un format plus facilement lisible
`-s`		Donne uniquement le total pour les dossiers
`-c`		Donne le total de la liste
`ln`	`destination lien`	(**l**i**n**k) Créer un lien dur
`-s`		Créer un lien symbolique plutôt qu'un lien dur

Caractères speciaux

`;`	Sépare des commandes à exécuter à la suite
`&&`	Sépare des commandes à exécuter à la suite (s'arrête si il y a eu des erreurs)
`~`	Désigne le chemin vers votre répertoire personnel (`$HOME`)
`.`	Désigne le répertoire courant
`..`	Désigne le répertoire parent
`?`	Symbole "joker", interprété comme « n'importe quel caractère »
`*`	Symbole "joker", interprété comme « n'importe quelle chaîne de caractère »
`\`	Inhibe l'interprétation du caractère spécial suivant
`'` et `"`	Délimite des chaînes de caractère
`` ` ``	Délimite une commande à interpréter dans les arguments d'une autre commande
`$VAR`	Désigne le contenu de la variable shell `VAR`
Exemples	
`echo "Je suis dans"; pwd`	Affiche du texte suivi du chemin absolu du répertoire courant
`compile prog.c && ./prog.exe`	Compile `prog.c` et exécute `prog.exe`, sauf si la compilation a échoué
`touch ~/toto`	Créé un fichier `toto` dans votre répertoire personnel
`touch ./toto`	Créé un fichier `toto` dans le répertoire courant (équivalent à `touch toto`)
`cd ..`	Va dans le répertoire parent au répertoire courant
`ls ../../rep2/file?`	Liste les fichiers `fileb` et `filec` dans `rep2`
`ls prog.*`	Liste les fichiers `prog.c` et `prog.exe` dans `rep1/sousrep1/`
`echo Une étoile : \*`	Affiche le texte « `Une étoile : *` »
`echo "Une étoile : *"`	Affiche le texte « `Une étoile : *` »
`` echo "Je suis dans `pwd`" ``	Affiche du texte suivi du chemin absolu du répertoire courant
`echo "J'habite dans $HOME"`	Affiche du texte suivi du chemin vers votre espace personnel

Gestion des entrées-sorties



`<`	Utilise un fichier comme entrée standard
`<<<`	Utiliser une chaîne de caractère comme entrée standard
`>`	Redirige la sortie standard vers un fichier (écrase l'ancien fichier)
`>>`	Redirige la sortie standard vers un fichier (à la suite du fichier)
`2>`	Redirige l'erreur standard vers un fichier
`&>`	Redirige la sortie et l'erreur standard vers un fichier
`|`	Redirige la sortie standard d'une commande vers l'entrée standard d'une autre commande
Exemples	
`./prog.exe < inputFile > outputFile`	Exécute `prog.exe` avec `inputFile` en entrée et met la sortie dans `outputFile` après l'avoir écrasé
`ls | ./prog.exe >> outputFile`	Exécute `prog.exe` avec la sortie de `ls` en entrée, et met la sortie à la suite de `outputFile`
`./prog.exe > /dev/null`	Exécute `prog.exe` en ignorant la sortie standard (`/dev/null` est l'équivalent d'une poubelle)

Raccourcis de la ligne de commande

TAB	Auto-compléter une commande ou un nom de fichier
TAB × 2	Montrer les possibilités d'autocomplétion (lorsqu'il y a ambiguité)
↑/↓	Parcourir les commandes de l'historique
`history`	Afficher l'historique des commandes
Ctrl + R	Rechercher une commande dans l'historique
Ctrl + A/E	Aller au début/à la fin de la ligne de commande
Ctrl + U/K	Supprimer tous les caractères à gauche/droite du curseur
Ctrl + W	Supprimer le mot à gauche du curseur

Gestion des processus

Ctrl + C	Arrêter l'exécution de la commande en cours
`commande &`	Exécute `commande` en tâche de fond
Ctrl + Z puis `bg`	Passe la commande en cours d'exécution en tâche de fond
`jobs`	Lister les processus de votre shell qui sont en tâche de fond
`ps -ef`	Lister **tous** les process en cours d'exécution sur la machine
`kill PID`	Tuer un processus en cours (via son identifiant `PID`)
`pkill processName`	Tuer un processus en cours (via son nom `processName`)
`-9`	Tuer brutalement le processus
`top`	Visualiser l'utilisation de la mémoire et du CPU par les processus
`-i`	Seulement les processus actifs
`-pPID`	Seulement le(s) processus correspondant(s) au(x) PID(s) donné(s)
`-uUSER`	Seulement les processus d'un utilisateur donné
`screen`	Ouvre une session screen qui permet de lancer des commandes dans un terminal qui continuera d'exister même si l'on quitte le terminal initial.

Lecture et édition de fichiers

`cat`	Affiche le contenu des fichiers dans la sortie standard
`less`	Lire et naviguer dans un fichier
`nano`	Edite un fichier avec `nano` (éditeur minimaliste)
`vim`	Edite un fichier avec `vim` (pour les ninjas)
`xemacs`	Edite un fichier avec `xemacs` (pour les pirates)
`nedit`	Edite un fichier avec `nedit` (en interface graphique)
`evince`	Lire un fichier pdf (en interface graphique)

Fichiers de configuration, variables d'environnement

`~/.bashrc`	Fichier de configuration qui est exécuté à chaque connexion
`env`	Commande qui liste toutes les variables d'environnement définies et leurs valeurs
`VAR=3.14`	Changer la valeur de `VAR` (pas d'espaces autour de `=`!)
`export VAR`	Rend la valeur de `VAR` disponible pour tous les process fils de ce shell
`echo "VAR vaut $VAR"`	Affiche la valeur de `VAR`
`$USER`	Nom d'utilisateur
`$HOSTNAME`	Nom de la machine
`$HOME`	Répertoire personnel
`$PATH`	Liste des répertoires où les commandes sont recherchées
`$LD_LIBRARY_PATH`	Liste des répertoires où les librairies sont recherchées
`$PS1`	Décrit la forme de l'invite de commande en bash
`$?`	Code de retour de la dernière commande lancée

Filtres et commandes avancées

Note : les filtres peuvent généralement être utilisés aussi bien sur un fichier que sur l'entrée standard (via « | »)

`grep`	Permet de filtrer ligne par ligne suivant un mot ou un motif
`sed`	Permet de remplacer une expression par une autre
`diff`	Afficher les différences entre deux fichiers
`cut`	Manipuler les colonnes d'une entrée
`tr`	Remplacer ou enlever des caractères
`find`	Rechercher des fichiers suivant des critères
`wc`	Permet de compter des mots, des lignes, ..
`bc`	Effectuer des opérations arithmétiques basiques
`ssh`	Se connecter de façon sécurisée à une autre machine
`scp`	Copier des fichiers entre deux machines
`tar`	Compresser ou décompresser des fichiers (format `tar`)
`gzip`, `gunzip`	Compresser ou décompresser des fichiers (format `gz`)
`alias`	Créer des alias

Exemples

`grep -nr "warning" ./`	Cherche récursivement les occurences de `warning` dans les fichiers du répertoire courant
`./prog.exe | grep "warning"`	Affiche seulement les lignes contenant `warning` de la sortie du programme `prog.exe`
`sed "s/search/replace/g" inputFile`	Affiche le contenu de `inputFile` en remplaçant toutes les occurences de `search` par `replace`
`diff file1 file2`	Compare les fichiers `file1` et `file2`
`cut -d " " -f 2,3 inputFile`	Affiche les colonnes 2 et 3 (`-f 2,3`) du fichier `inputFile`, par rapport aux espaces (`-d " "`).
`cat inputFile | tr " " ":"`	Affiche le contenu de `inputFile` avec les espaces remplacés par `:`
`find ../ -name "*.cpp"`	Trouver tous les fichiers se finissant par `.cpp` dans le répertoire parent
`wc -l *`	Compte le nombre de ligne pour chaque fichier du répertoire
`bc <<< "1+2"`	Effectue l'opération 1+2 et affiche le résultat
`ssh user@host`	Se connecter à la machine `host` en tant que `user`
`scp user@host:/home/user/file ./`	Copie le fichier `/home/user/file` situé sur la machine `host` dans le répertoire courant
`lpr -Pbb8 outputFile`	Imprime le fichier `outputFile` avec l'imprimante nommée `bb8`
`tar -xvf archive.tar`	Décompresse le fichier `archive.tar`
`gzip inputFile`	Compresse `inputFile` en un fichier `inputFile.gz`
`alias ll='ls -l'`	Créer un alias `ll` qui correspond à la commande `ls -l`