Formation “Prise en main de l’outil informatique”

Jean Pommier, jean.pommier@pi-geosolutions.fr

IdGeo, 3 décembre 2024

Sources : https://github.com/pi-geosolutions/formation_prise_en_main_outil_informatique

---

Ce cours a pour objectif d’homogénéiser votre niveau de maîtrise de l’outil informatique de base. Pour des raisons temporelles, nous ne pourrons pas tout couvrir ni aller très loin sur chaque sujet. L’idée est de vous donner quelques bases, et peut-être l’envie de creuser certains sujets.

Il est probable que pour certains d’entre vous ce cours soit trop lent, dénué d’intérêt voire ridicule. Presque à coup sûr, le site d’entraiment dactylo que nous verrons au début pourra vous occuper pendant les temps morts si besoin.

Cependant, on peut aussi voir les choses comme suit :

- parmi toutes les choses que vous savez déjà, il y en aura certainement qui ont jusque là échappé à votre attention. Ce serait dommage de les rater
- votre voisin.e est peut être moins à l’aise avec le sujet en cours, peut-être pouvez-vous l’aider ?
- sur les sujets où vous en savez plus que le prof, partagez vos connaissances. Si on apprend tous, c’est encore mieux ! Sans non plus totalement perturber le cours, si on peut...

Ordre proposé:

1. clavier
2. windows
3. linux
4. introduction au concept de base de données

Savoir utiliser son clavier

Durée

1 à 2h

Objectifs

- ☐ Connaître les touches et usages Notamment tabulation, caps lock, shift, Alt Gr
- ☐ Notions de dactylo
  - on est supposés ne pas avoir besoin de regarder le clavier quand on tape
  - positionner ses dix doigts sur le clavier
  - application d’apprentissage dactylo (https://www.typingclub.com/)
  - utiliser au moins 4 doigts pour frapper au clavier

- ☐ Raccourcis clavier
  - ☐ connaitre les raccourcis claviers courants
  - ☐ savoir se débrouiller sans la souris
    - lancer le navigateur internet, passer d'un onglet à l'autre, lancer une recherche
    - éteindre l'ordi

Exercices

1. Initiation dactylo

Aller sur https://www.typingclub.com/ et suivre les premières leçons

Ca donne de bonnes bases. Si vous avez un temps mort durant les cours, vous pourrez toujours y revenir pour améliorer votre maitrise.

2. Raccourcis clavier

Quels sont les raccourcis pour ?

- Edition
  - Copier
  - Coller
  - Couper
  - Tout sélectionner
  - Annuler la dernière action
  - Restaurer la dernière action
  - Enregistrer un fichier
  - Enregistrer un fichier et définir le nom
  - Texte en gras
  - Texte en italique
  - Chercher dans le texte
  - Remplacer dans le texte
- Actions sur les fichiers / fenêtres
  - Ouvrir un fichier
  - Créer un nouveau fichier
  - Afficher les propriétés d'un fichier
  - Quitter l'application
  - Ouvrir le menu Windows
  - Passer à l'élément suivant
  - Afficher/masquer le bureau Windows
- Dans le navigateur
  - Ouvrir dans un nouvel onglet
  - Fermer un onglet
  - Restaurer un onglet fermé
  - Restaurer une fenêtre fermée
  - Recharger la page
  - Revenir à la page précédente (historique-arrière)

3. Savoir se débrouiller sans la souris

Fermez toutes vos fenêtres, on part d'un environnement vide, propre. Débranchez la souris. On va s'en passer.

- Avec uniquement le clavier, démarrez votre navigateur internet
- Ouvrez un onglet, faites une recherche
- Ouvrez un résultat de recherche dans un nouvel onglet
- Revenez à l'onglet de recherche.
- Fermez l'onglet
- Fermez le navigateur
- Redémarrer l'ordi

Sujet suivant : windows

Windows

Durée

2 à

Objectifs

- ☐ raccourcis clavier classiques : cf sujet clavier
- ☐ Arborescence système
  - ☐ bonnes pratiques
  - ☐ trouver un fichier
  - ☐ créer un fichier/dossier
    - avec la souris
    - avec le clavier
  - ☐ Connaître la correspondance de nommage entre les raccourcis dans l'explorateur et les dossiers a la racine du système (où va quoi ?)
- ☐ Editeur de code
  - ☐ choisir
  - ☐ configurer
  - ☐ savoir utiliser.
  - ☐ Raccourcis clavier
- ☐ navigateur internet
  - ☐ choisir son navigateur
  - ☐ recherche vs URL
  - ☐ configuration (téléchargement)
  - ☐ outils de développement -> CTF dans des fichiers HTML
- ☐ Scripts
  - ☐ script .bat ? -> peut être pas
  - ☐ utiliser powershell (base)
    - ☐ naviguer dans le FS
    - ☐ lire / créer fichier
    - ☐ traitement en boucle

Editeur de code - Notes de cours

Pour travailler sur des fichiers bureautiques, on utilise en général une suite bureautique (OpenOffice, LibreOffice, MSOffice, etc). Pour le code, il en va de même, on utilise un éditeur de code.

On peut utiliser soit:

- un éditeur de code simple et polyvalent. C'est la solution la plus simple, mais parfois un peu limitée. Parmi les éditeurs gratuits sous Windows, on peut citer notepad++ et Visual Studio Code ou sa version open source VSCodium.
- une IDE (*integrated development environment*) qui fournit une interface plus riche, avec des assistants, raccourcis etc. Tout un environnement dédié en général à un langage, framework ou flux de développement. Une IDE permet en général de travailler plus efficacement mais nécessite un temps de prise en main.

Dans le cadre de ce cours, on va se focaliser sur la première catégorie, les éditeurs de code génériques. De toutes façons, ça sert toujours.

Editeur de code, editeur de texte, quelle différence ? On voit parfois des gens essayer d'éditer du code dans Wordpad. C'est ridicule, tout développeur s'accordera à le dire. Ou, mais pourquoi ?

Wordpad = mauvaise idée !. Applique un formattage au texte. Il s'apparente plutôt à un Word allégé. Les tabulations en particulier ne seront pas traitées correctement. Les guillements, aussi, seront sans doute remplacés. Bref, ça va pourrir le code. Il est peu probable qu'il marche après édition, s'il marchait avant.

OK. Le Bloc-note alors ? Bof bof C'est plus acceptable. Mais particulièrement inefficace.C'est un outil grand public, pas du tout pensé pour le code.

Un éditeur de code fournira plein d'outils qui vont vous faciliter la vie. Revenir en arrière est impossible. Alors, pillule bleue ou pillule rouge ?

Choisir un éditeur de code Il y a un apprentissage et chaque éditeur fait à sa façon. Vous avez le choix de l'éditeur. Mais essayez de vous fixer sur un et investir assez de temps pour le maitriser.

Dans ce cours, on va couvrir notepad++ et Visual Studio Code.

Chacun de ces logiciels propose des fonctions de base, déjà bien riches, que vous pouvez compléter à l'aide de plugins :

- liste des plugins pour notepad++
- liste des plugins pour VS code

Quelques fonctionnalités utiles

- complétion automatique : selon le langage utilisé, les mots clefs du langage vous seront proposés. Vous pouvez les valider avec la touche Tab.
- Commenter des lignes : `Ctrl+/` pour VS code, `Ctrl+Q` pour np++
- Indentation : l'indentation est le décalage par rapport à la gauche de votre texte. Certains langages comme Python structurent leur code via l'indentation. Et pour les autres, ça permet de rendre le code lisible. C'est fondamental.
- sélection/édition en colonne : `Alt+Shift` pour les deux.

Editeur de code - Exercices

1. Arborescence système

- Créer un dossier pour les données liées à ce cours, à l'emplacement prévu par les bonnes pratiques idgeo
- Y créer un fichier pour vos prises de notes. Je vous propose un fichier avec extension .md (Markdown). On verra pourquoi dans la partie dédiée à l'éditeur de code.

2. Arborescence système

- Où est stocké réellement le contenu du dossier Documents ?
- Où vont les fichiers stockés sur le Bureau ?
- Quel est le dossier système de Windows ?
- Où sont installés les logiciels ?

Pour éclairer notre gouverne, on va prendre un peu d'avance sur le cours et ouvrir un terminal powershell.

- Tapez la commande `ls C:/`. Des commentaires sur ce que vous voyez ?
- Tapez la commande `ls C:/Users`. Des commentaires sur ce que vous voyez ?

3. Configurer son éditeur de code

Config de base

Panneau latéral Quand on code, on travaille sur plusieurs fichiers. Les ouvrir un par un est particulièrement pénible. Et parfois, on ne sait plus trop lequel est ouvert. Le panneau latéral est là pour ça.

Notepad++: Affichage -> Projet -> Panneau de projet 1. Et puis on y ouvre les dossiers qui nous intéressent

VS code: Barre à gauche, première icône. Et puis on y ouvre les dossiers qui nous intéressent

Coloration syntaxique Par ce terme, on entend la mise en évidence des mots clef d'un langage de programmation. Ca facilite grandement la lecture du code. Essayez donc d'ouvrir ce fichier dans votre éditeur et d'obtenir la coloration syntaxique propre au langage Markdown.

Prévisualisation markdown Tant qu'on y est, votre éditeur propose un outil de prévisualisation du code Markdown (affichage propre imprimable).

Il est là par défaut pour VS code. Pour np++, il faut installer un plugin : MarkdownPanel

Configuration des tabulations Plusieurs conventions existent, mais en général dans le code, on va vouloir qu'une tabulation corresponde à 2 ou 4 vrais espaces (une tabulation peut aussi être un caractère spécial). Souvent la valeur par défaut est 4 espaces. On va vérifier ça, et changer pour 2 espaces.

Multi edit VS code le propose d'office avec la touche `Alt`.

Pour np++, il faut l'activer sur Paramètres -> Préférences -> Zones d'édition

4. Utiliser son éditeur de code

- Ouvrir le fichier XML. La coloration est elle bien activée ? Remplacer toutes les balises `<Utilisateur>` par des balises `<User>`
- Dans ce même fichier XML, un `s` s'est glissé à la fin des prénoms aux lignes 11, 20 et 24. Supprimez ces trois `s` *en une seule édition*
- Ouvrir le fichier CSV. Aux lignes 5 à 24, pour la colonne `CDBIKESTATIONID` remplacer la valeur par `ECO035`.
- Ouvrir le fichier python (`.py`). Le bloc de la fonction `blablah` ne marche pas, il n'est pas indenté comme il faut. Corriger cela.
- Commenter des blocs de code :
  - dans le fichier XML
  - dans le fichier SQL
  - dans le fichier python

Navigateur Internet

Choisir son navigateur

Vous avez l'embarras du choix :

- Microsoft Edge
- Mozilla Firefox
- les navigateurs basés sur le moteur chromium
  - Chrome
  - Brave
  - ...
- Opera

et sans doute d'autres encore.

On évitera Edge. S'il marque un net progrès par rapport aux précédents navigateurs Microsoft, il pose encore des problèmes en termes de support des standards.

De manière générale, je dirais que vous pouvez bien choisir celui qui vous plait. Mais apprenez à l'utiliser correctement. Et ne subissez pas le navigateur par défaut de votre ordi, choisissez le vôtre, que vous connaissez.

Nous allons considérer l'alternative Firefox / Chrome, qui doivent être déjà installés sur votre ordi.

Recherche vs URL

Depuis quelques années, on peut saisir sa recherche google directement dans la barre d'URL. Cette simplification en termes d'UI (interface utilisateur) est discutable en termes d'UX (expérience utilisateur) :

- ce que vous tapez dans la barre d'url est automatiquement envoyé au moteur de recherche (et donc enrichit les stats d'usage)
- si vous faites une erreur en tapant une URL et que celle ci n'est pas valide, cela va automatiquement se transformer en recherche, et vous pourrir votre saisie
- et ceci est particulièrement agaçant quand on développe des applis web en local

Personnellement, j'aime garder la séparation. Mais ça devient de plus en plus difficile.

Sous firefox : - dans la barre d'url, taper `about:config` - chercher le mot clef `handoff` et passez `browser.newtabpage.activity-stream.improvesearch.handoffToAwesomebar` à false. Ca évitera que le navigateur tape votre recherche dans la barre d'url. - de la même manireè, passer à `false` les params suivants : - `browser.urlbar.suggest.searches` - `keyword.enabled` - `browser.fixup.alternate.enabled` - dans les paramètres de configuration (about:preferences), on pourra agir sur - Recherche -> ajouter la barre de recherche - Vie privée -> Barre d'adresse et désactiver les moteurs de recherche (suggestion uniquement)

Sous Chrome : Dans les préférences : - Services Google/synchronisation, on peut désactiver "Améliorer les suggestions de recherche"

En termes de protection des données et de la vie privée, c'est déjà ça. *Mais je n'ai pas trouvé de moyen de désactiver la recherche dans la barre d'adresse...*

Configuration

Dans les options de configuration, ce n'est pas fondamental, mais un certain nombre d'options super pratiques ne sont pas actives par défaut :

- **Sessions** : à l'ouverture du navigateur, restaurer les onglets ouverts de la session précédente
- **Téléchargements** : le lieu où les téléchargements sont sauvegardés : j'aime contrôler cet aspect, et donc choisir à chaque fois au lieu de subir un emplacement automatique

Outils de développement

C'est un outil "avancé", mais tellement pratique. On peut l'ouvrir via le menu contextuel (clic droit), ou bien la touche `F12`. On prend un peu d'avance sur d'autres cours. Mais ça ne peut pas faire de mal.

Il offre divers outils, globalement les mêmes qu'on soit sous Firefox ou Chrome, possiblement nommés différemment. Je vais ici citer Firefox.

- **Inspecteur** : affiche le code HTML/CSS d'une page web. On peut y trouver des infos intéressantes, pratiques, expérimenter des changements dans le layout de la page, très pratique quand on code une page web ou du style. Parfois aussi quand on veut récupérer une information grossièrement cachée
- **Console** : console javascript. Peut permettre d'exécuter du javascript, d'inspecter des variables javascript. Affiche aussi les logs javascript. Pratique notamment en cas d'erreur sur une page, pour comprendre ce qui se passe.
- **Débogueur** : usage plus avancé avec le code javascript
- **Réseau** : un de mes outils favoris. Liste toutes les requêtes réseau effectuées par la page. Très pratique pour identifier du traçage. Mais aussi pour comprendre ce qui ne marche pas sur une page, ou identifier les URLs de services utilisés
- les autres onglets, on ne va pas regarder

Navigateur Internet – exercices

On déroule https://github.com/pi-geosolutions/enigmes_html

Eventuellement, on enchaine sur les premières de http://ouverture.pas.facile.free.fr/

Scripting avec Windows – notes

Windows a fait son succès initial avec les interfaces graphiques et l'usage de la souris. Au point que parler de scripting, de console et de clavier semble réservé aux hackers. Et d'ailleurs, les hackers qu'on voit à la télé, si on regarde bien leur écran, font souvent bien rigoler...

Il faut dire que les outils pour écrire du script, dans Windows... découragent le curieux.

Malgré tout, parce que nous sommes braves, téméraires et qu'il le faut bien, nous allons commencer par regarder comment ça se passe avec Windows. Et ensuite, on verra comment se simplifier la vie...

DOS ou powershell ?

DOS : Le DOS existe depuis "toujours". Et ça se voit : peu ergonomique, très limité, il ne fait pas envie. Malgré tout, il existe encore, et même, vous allez sans doute le voir passer de temps en temps cette année, c'est par exemple parfois lui qui est fourni préconfiguré avec QGIS.

De base, il s'ouvre en tapant `cmd` dans l'invite Windows. A l'occasion, on ira y faire un tour.

Powershell : c'est son remplaçant, afin d'offrir à Windows un shell digne de ce nom. Notamment pour les usages sur serveur informatique, où l'interface graphique n'est souvent même pas une option... et où Linux se taille la part du lion.

Bien plus puissant que son ancêtre, il permet de réaliser des boucles, de chaîner des commandes, il offre aussi la complétion automatique (finit la commande lorsqu'on appuie sur la touche `Tab`), bref, c'est mieux. Et il reconnait un certain nombre des commandes de bases des shells courants de linux.

Shell : c'est comme ça qu'on appelle l'interface utilisateur en ligne de commande. Powershell sous Windows, bash, sh, zsh sous Linux sont des shells. Bon, pour DOS, je ne me prononcerai pas.

Commandes Powershell de base

Naviguer dans le système de fichiers Par défaut, il démarre à l'emplacement de notre compte utilisateur. Le chemin est précisé juste avant l'invite de commande. Quelque chose vous frappe ?

Le système de fichiers n'est pas vraiment vu de la même façon que via la GUI (interface graphique utilisateur). La GUI applique des modifs cosmétiques. Mais qui compliquent vite les choses. Avec le shell, vous êtes de l'autre côté du miroir.

Un bon nombre des commandes listées ici sont reprises du shelle bash (linux), on les retrouvera donc sans surprise, mais plus puissantes, quand on verra linux.

qq commandes :

- `pwd` : savoir où on est (chemin absolu)
- `cd` : se déplacer (`cd` = *change directory*).
  - Soit de façon relative :
    - `cd ..` (remonter d'un niveau),
    - `cd mon-sous-dossier` (descendre d'un niveau),
    - `cd ../..` (on peut faire plusieurs étapes d'un coup)
  - Soit de façon absolue : `cd c:/Users/`

  - On peut aussi spécifier des chemins avec espaces, même si c'est moins pratique, et pas supporté partout : `cd 'C:/Documents and Settings/'`
  - On notera qu'il y a une indifférence à la casse (lettre majuscules ou minuscules)
- `ls` : lister les fichiers
  - là où on est : `ls`
  - à un chemin donné : `ls C:/Windows`
  - lister les éléments correspondant à un motif : `ls *.csv`

Lire, copier, supprimer un fichier/dossier

- `mkdir` : créer un dossier.
  - `mkdir dossier1`
  - `mkdir -p dossier1/sous-dossier`
- `cp` : copier un fichier. `cp source.csv destination/csv`
- `rm` : supprimer un fichier/dossier.

Il est parfois pratique de n'afficher que le début (ou la fin) d'un énorme fichier. Là ou le charger dans un editeur de code sera long et gourmand, en ligne de commande ça devrait être quasi instantané

- `gc log.txt | select -first 10` affiche les 10 première lignes
- `gc -Tail 10 log.txt` affiche les 10 dernières lignes
- `gc log.txt | more` # or less if you have it installed
- `gc log.txt | %{ $_ -replace '\d+', '($0)' }` chercher-remplacer

Boucles La syntaxe de base est

```
for (<Init>; <Condition>; <Repeat>) {

<Script Block>

}
```

Par exemple :

```
for ($var = 1; $var -le 5; $var++) {
    Write-Host The value of Var is: $var
}
Write-Host End of for loop.
```

Ou pour une boucle sur une liste de chaines de caractères :

```
$employees = @("Bijay","Bhawana","Padmini","Lakshmi")
foreach ($emp in $employees) {
  $emp;
 }
```

Ou un exemple plus avancé, où l'on découpe avec ogr2ogr toute une série de données geospatiales :

```
foreach($file in Get-ChildItem midi-pyrenees-osm/*.shp) {
    ogr2ogr.exe "$($file.BaseName)_09.shp" -clipsrc ../departement-ariege.shp -lco ENCODING=UTF-8
}
```

Terminal Windows

Dans le MS Store, vous trouverez une application appelée Windows Terminal. Elle est plutôt sympa, elle vous permet de lancer plusieurs terminaux de ligne de commande dans des onglets, comme le navigateur. Et pour chacun, de choisir entre les shells installés : powershell, DOS, linux (voir plus loin) etc

Scripting avec Windows – exercices

- créer un dossier `tmp` à la racine du disque D
- dans ce dossier `tmp`, créer un dossier `shell/data`
- copier le fichier csv du dossier partagé dans ce dossier
- en fait, on va copier plusieurs fichiers d'un coup
  - supprimer le fichier csv qu'on vient de copier
  - copier le dossier sample-data dans le dossier tmp/shell/data
- lister le contenu du dossier
- afficher à l'écran les 10 premières lignes du fichier CSV
- maintenant, imaginons que nous voulons préparer un espace de travail pour plusieurs stagiaires, un dossier par stagiaire. Les stagiaires s'appellent Arthur, Marwanne, Lucie, Jérôme et Bjorn.
  - créer un dossier destination tmp/shell/stagiaires
  - dans ce dossier, créer un dossier pour chacun des stagiaires, lui-même contenant les sous-dossiers `cours`, `exercices`, `scripts`, `tmp`

Qu'est-ce qu'on s'amuse, hein ?

Sujet suivant : linux

Linux

Durée

2 à 3h

Objectifs

- ☐ WSL
- ☐ Arborescence système
- ☐ Installer des logiciels
- ☐ lignes de commande, bash
- ☐ Scripts
  - ☐ boucles
  - ☐ chercher un fichier
  - ☐ grep
  - ☐ sed
  - ☐ écrire un fichier de script exécutable

WSL2

Il ne s'agit pas à proprement parler de Linux. Mais d'une façon de faire tourner un système linux dans Windows. Enfin, à ce que j'ai compris, plutôt une émulation, une sorte de machine virtuelle. On n'a pas tout, loin de là. Mais c'est assez bien intégré dans Windows. Et c'est très pratique.

Windows depuis la version 10 permet d'installer un noyeau linux, dans son système Windows, ce qui permet de disposer d'un système linux en ligne de commande (pas d'interface graphique, par défaut) qui peut interagir avec le système Windows. Ca s'appelle WSL (Windows Subsystem Linux).

Installer WSL2

Sur les ordis IDGeo, ça devrait déjà être fait.

Mais sinon, suivre les instructions : https://learn.microsoft.com/fr-fr/windows/wsl/install

Comme le dit la doc, par défaut je crois qu'il installe une ubuntu. Si on veut autre chose, il faut le spécifier.

Installer une distribution linux

Une fois WSL activé, par défaut je crois qu'il installe une ubuntu. Si on veut installer une distrib additionnelle par exemple, on peut passer par le Windows Store. Si si.

Spécificités de WSL

Je vais sûrement en rater, mais en vrac :

- pas d'interface graphique linux, uniquement ligne de commande
- depuis linux, le système de fichiers Windows s'accède via un chemin typiquement linuxien, dans /mnt. Ex. : C:/ se voit depuis le système linux à /mnt/c/
- depuis Windows, le système de fichier linux se trouve dans `\\wsl$\`. Par exemple, mon compte utilisateur dans ma debian se trouvera à `\\wsl$\Debian/home\jean`

Arborescence système Linux

Ca dépend un peu de la distribution utilisée. On va parler ici de Debian/Ubuntu. TODO: Ecrire un résumé

Installer des logiciels

Ca dépend un peu de la distribution utilisée. On va parler ici de Debian/Ubuntu.

Linux utilise un système de "stores" depuis bien avant que ça devienne la mode. La plupart des logiciels et librairies supportées par une distribution sont listées dans un grand magasin d'applications, auquel on accède de manière unifiée. On peut ajouter des dépôts pour agrandir la liste si besoin. Ensuite, un outil unique permet de chercher des entrées dans ce magasin, d'installer des logiciels, de faire des mises à jour, etc.

Par exemple, si je veux utiliser la librairie GDAL/OGR dans linux, je vais juste demander à l'installer, et elle sera directement disponible sans complication, variable d'environnement ou quoi :

```
# sudo passe en mode admin le temps de la commande.
# apt est l'outil de gestion du "store".
# update : recharge la liste des applications
sudo apt update

# Si on ne connait pas le nom exact, on peut faire une recherche
sudo apt-cache search gdal

# Et si on le connait, c'est aussi simple que ça :
sudo apt install gdal-bin

# On vérifie que ça marche. Affichons l'aide d'ogr2ogr
ogr2ogr --help
```

Bash

Bash est le shell le plus répandu dans l'environnement linux. Enfin, c'est faux, c'est sh mais il est moins riche.

Vous allez retrouver à peu près toutes les commandes listées pour le powershell, puisqu'elles viennent de bash. Avec plus d'options et de puissance, vous aurez : pwd, cd, ls, mkdir, cp, rm.

Pour afficher le contenu d'un fichier, ça sera

- `head log.txt` pour les 10 premières lignes
- `tail log.txt` pour les 10 dernières

Pour créer un fichier : `touch mon_fichier.md`.

Pour envoyer le résultat d'une commande dans un fichier : `head -n 50 users.csv > users_sample.csv` va créer un fichier limité aux 50 premières lignes.

Télécharger un fichier ? Utilisez `wget` ou `curl`.

Editer un fichier ? `nano` pour les utilisateurs occasionnels, `vim` ou `emacs` pour les ambitieux.

Chercher

- `grep` pour chercher dans un fichier :
  - `grep jean users.csv`
  - `grep -R "jean" .` : cherche dans tous les fichiers à partir du dossier courant
- `sed` pour remplacer :
  - `sed -i "s/jean/jpommier/" users.csv` : chercher/remplacer jean par jpommier
- `find` pour trouver un fichier
  - `find . -name user*.csv` : chercher tous les fichiers user*.csv à partir du dossier courant

Boucles

Par rapport au Powershell, la syntaxe est un peu plus simple :

```
for i in {1..5}
do
```

```
    echo The value of i is: $i
done
```

Ou pour une boucle sur une liste de chaines de caractères :

```
employees="Bijay Bhawana Padmini Lakshmi"
for emp in $employees
do
  echo $emp;
done
```

Et enfin, pour reprendre l'exemple précédemment donné :

```
for f in $(ls -Sr midi-pyrenees-osm/*.shp)
do
    ogr2ogr $(basename -- ${f%.*})_09.shp -clipsrc ../departement-ariege.shp -lco ENCODING=UTF-8 $
done
```

Ecrire un fichier de script exécutable

De base, on tape ces commandes et on les enchaîne depuis le terminal.

Mais ensuite, si on veut reproduire ces actions, il est préférable de les organiser dans un fichier de script. Un fichier de script peut vite devenir sa propre documentation

Un fichier bash doit :

- commencer par la ligne `#!/bin/bash`
- être exécutable

On peut écrire des commentaires en les débutant par `#`

On peut définir des variables (Ex. `ma_var=quelquechose`) et l'appeler avec $ (ex. `echo $ma_var`)

On peut écrire des fonctions pour les opérations répétitives, ou pour structurer le code

Tâches répétitives

Si vous avez les commandes à exécuter à intervalles régulier, vous pouvez utiliser `cron`.

Bash – exercices

Ecrire un script

Supposons que dans le cadre d'un projet lié au prix du foncier, nous voulons produire un fichier recensant les prix du foncier pour un lot de communes autour de Colomiers. Un fichier par commune.

Notre donnée source sera https://www.data.gouv.fr/fr/datasets/demandes-de-valeurs-foncieres-geolocalisees/

La liste des codes insee des communes : 31032, 31056, 31069, 31088, 31149, 31150, 31157, 31291, 31351, 31417, 31424, 31526, 31555, 31557

Première approche En première approche, on va se simplifier la vie : le service fournit des jeux de données par commune.

Vous allez créer un script qui

- crée un dossier principal et dans ce dossier, un sous-dossier par commune
- télécharge les fichiers pour chaque commune, les stocke dans le sous-dossier correspondant et le décompresse
- Utilise ogr2ogr pour transformer le fichier CSV en fichier geopackage

Deuxième approche Supposons que le service ne fournisse pas des extractions par commune et qu'on doive faire avec un plus gros fichier.

Vous allez créer un script qui

- crée un dossier principal et dans ce dossier, un sous-dossier par commune
- télécharge le fichier CSV du département Haute Garonne (31) et de dézippe
- copie le template VRT fourni ici dans chacun des dossiers de commune
- remplace la chaine de caractère REMPLACER_PAR_CODE_INSEE_COMMUNE dans chacune des copies du VRT, afin de l'adapter à chaque commune
- Utilise ogr2ogr pour transformer la source VRT en fichier geopackage

Note 1 : VRT est un format inveté par GDAL/OGR, qui permet de définir une donnée virtuelle à partir d'une donnée source et d'un fichier .vrt qui définit une transformation à appliquer à cette donnée source. Vous aurez un cours dédié à ça durant votre cursus. On ne va donc pas rentrer trop dans les détails. Mais pour les impatients : https://gdal.org/drivers/vector/vrt.html.

Note 2 : bien sûr, on pourrait faire plus simple. Mais on essaie d'appliquer ce qu'on a vu en cours.

Linux et les serveurs informatiques

Dans l'ensemble, on peut définir un serveur comme un ordinateur connecté au réseau, fournissant des services distants et auquel on accède sans interface graphique.

Attention, Linux peut fournir une interface graphique. Je dirais même qu'il est meilleur que Windows pour ça. en tous cas il offre plus de choix. Mais là où il se démarque vraiment, c'est quand on n'a *pas* d'interface graphique.

Il nous reste alors la ligne de commande. Nous y sommes.

Connexion distante

Un outil omniprésent lorsqu'on parle de connexion distante : `ssh` (Secure Shell). Il permet de se connecter en ligne de commande à une machine distante, de façon totalement sécurisée. On y est alors comme chez soi... en ligne de commande

Tunnel SSH

On peut même faire un pas de plus, et jouer à saute-mouton : ssh permet beaucoup de choses, notamment d'ouvrir des *tunnels*. Un tunnel permet de passer par un serveur distant pour se connecter à une autre machine à laquelle seul ce serveur distant pouvait se connecter. Sauf qu'avec le tunnel, une fois établi, c'est comme si cette autre machine était en local !

Note : on peut aussi faire du ssh et même du tunneling depuis Windows, certains outils permettent ça, notamment le très connu `Putty`. Mais bon... comment faire mal, en 20 clics ce qu'une seule ligne de commande fait bien et proprement ? Ca s'eppelle en 5 lettres, ça commence par 'P'.

Sujet suivant : bases de données

Introduction au concept de bases de données

Durée

?

Objectifs

- ☐ Utiliser Excel comme un SGBD
  - ☐ différence colonnes/lignes
  - ☐ pas de mise en page
  - ☐ schema de données
  - ☐ rendu formatté : tableau croisé dynamique
- ☐ CSV, un format d'échange
- ☐ ogr2ogr et VRT, présentation rapide
- ☐ Premier contact avec un vrai SGBD
  - ☐ on pousse les données excel dans une BD postgresql (ogr2ogr + vrt, je le fais moi ou je les guide suffisamment)
  - ☐ requêtes SQL (ou comment faire du tableau croisé dynamique sans souris)
  - ☐ insertion

Tableur : base de données ou pas ?

Tableur = MS Excel, Libreoffice Calc etc.

Les puristes vous diront que non, ce n'est pas une base de données. Dans un sens, c'est vrai. En tous cas pas une base de données relationnelles.

... sauf que c'est malgré tout l'outil que la plupart des gens utilisent pour inventorier, organiser et stocker leurs données. Et les éditer.

Alors ça vaut le coup d'en parler

- pour mieux intégrer ces pratiques dans un usage de base de données. Notamment structurer proprement les données dans les feuilles de votre tableur
- avoir connaissance de méthodes permettant de laisser les gens travailler sur leur tableur, pour un usage final en base de données
- essayer de mettre en place un flux performant qui sortira le moins possible vos collègues de leurs habitudes de travail. Tout en les aidant à mieux se servir de leurs outils.

Ne pas faire...

A *ne pas* faire :

- **fusion de cellules** : ***à éviter absolument***, pour un usage de type BD. On en a tous fait, et c'est OK d'en faire quand il s'agit d'un petit tableau, destiné à aller dans un rapport, des slides ou autre. Mais pour stocker de la *donnée*, c'est une très mauvaise idée :
  - On ne peut plus faire de tri
  - On ne peut plus faire de filtre
  - On ne peut plus faire de tableau croisé dynamique
  - On ne peut plus faire de formule
  - On ne peut plus faire de macro
  - Parce que ça va énerver celui qui récupérera votre travail et qu'il ne pourra rien en faire
  - Parce qu'il suffit de 3 clics avec un tableau croisé dynamique pour créer un joli rapport avec des cellules fusionnées automatiques
- **sauts de ligne** : sauter une ligne, c'est pareil, c'est tentant pour aérer et structurer le contenu. Mais c'est pénible à gérer ensuite. Et, par exemple, ça risque de casser le bon fonctionnement d'un tableau croisé dynamique. Mieux vaut jouer sur la couleur, ou bien la hauteur de la ligne
- **Changer de colonne pour un niveau inférieur** : une même donnée doit rester dans la même colonne. On voit régulièrement des tableaux où une indentation permettant de distinguer plusieurs niveaux se fait par la changement de colonne. Inutilisable en mode données.
- **Ajouter une colonne pour chaque nouvelle année/date/thématique**. Comme nos écrans sont plus larges que haut, la tendance la plus intuitive serait de rajouter des *colonnes* plutôt que des *lignes*. C'est vite difficilement utilisable. En principe, le nombre de colonnes ne devrait pas changer dans le temps. C'est les lignes, qui s'incrémentent.

Exemples

- 1 colonne par année : https://www.data.gouv.fr/fr/datasets/donnees-de-comptabilite-generale-de-letat/, fichier 2012-2022-balances-des-comptes-de-letat.csv
- joli tableau mais cellules fusionnées (et 1 colonne par année) : https://www.data.gouv.fr/fr/datasets/donnees-de-comptabilite-generale-de-letat/, fichier 2006-2022-bilan-cdr-solde.xlsx 2e feuille
- colonnes répliquées : https://opendata.lillemetropole.fr/explore/dataset/menu-cantine/table/

Bonnes pratiques

- noms des colonnes dans la première ligne
- données dans les lignes suivantes
- attention au *type* (ou *format*) des données

Exemples

- https://opendata.lillemetropole.fr/explore/dataset/wwwroubaixshopping/table/
-

Si on veut faire une synthèse des données

- tableau croisé dynamique
- graphique
- à la main avec des fonctions comme `NB.SI.ENV`, `SOMME.SI.ENV` etc

https://documentation.libreoffice.org/assets/Uploads/Documentation/en/CG7.1/CG71-CalcGuide.pdf

CSV, un format d'échange

https://en.wikipedia.org/wiki/Comma-separated_values

On va rarement travailler sur un fichier CSV. Par contre, on va souvent en rencontrer, par exemple sur https://data.gouv.fr.

CSV = *Comma Separated Values*

En pratique, on pourra trouver différents types de séparateurs. En France, ça sera souvent le `;`.

Il existe aussi une variante, moins courante, le TSV (tabulation)

CSV est un format *texte*. C'est à dire que vous pouvez l'afficher facilement avec n'importe quel outil. Un exemple simple : `head monfichier.csv` permet de voir la structure et un aperçu du contenu du fichier CSV, rapidement même si celui-ci est énorme, dans votre console. Si on parle d'une donnée de plusieurs millions de lignes, la charger dans votre tableur sera une autre affaire. A comparer avec un format xlsx ou ods, qui ne pourra pas se visualiser facilement hors d'un tableur. Ou s'éditer.

Tout logiciel et tableur sera capable de lire du CSV (et permet de configurer les paramètres d'import, comme le séparateur). Il permettra aussi d'exporter en CSV. C'est donc un format d'échange bien pratique.

Son principal inconvénient : il ne stocke pas les types des colonnes (Texte, Entier, Date etc).Il existe bien quelques essais pour corriger ce souci (fichier CSVT (QGIS, GDAL/OGR), Tabular data package, type prefixes). Mais l'adoption est encore assez limitée.

Dans un usage geospatial avec QGIS et GDAL/OGR, on pourra s'intéresser

- aux fichiers .csvt, voire au GeoCSV
- ou bien au format VRT, qui a l'avantage de couvrir bien plus que le simple format CSV

VRT ? GDAL/OGR ?

https://gdal.org/

GDAL/OGR est un projet open source, qui fournit :

- une librairie logicielle (brique à assembler dans un programme) d'entrée/sortie pour de nombreux formats. En clair, elle permet de lire et d'écrire de très nombreux formats. La plupart sont goepspatiaux. Certains sont plus génériques, comme par exemple CSV, XSL(X), ODS. La liste est fournie sur leur site : formats vecteur, formats raster
- un ensemble de programmes et de scripts permettant de manipuler les données supportées.

La librairie GDAL/OGR est utilisée un peu partout dans l'univers des SIG, et pas que dans l'open source. Sa licence assez souple (MIT) permet son utilisation dans des projets propriétaires. Elle est par exemple utilisée, à ma connaissance, dans ArcGIS Desktop et FME. Pas mal, non ?

GDAL concerne les données raster (images)

OGR concerne les données vecteur (géométries vectorielles, données tabulaires)

Nous allons donc plutôt ici regarder du côté d'OGR

VRT

https://gdal.org/drivers/vector/vrt.html#vector-vrt

J'ai découvert le format VRT récemment, lors du Geocom 2021 (conférence de la communauté geOrchestra). Le seul à s'être tenu en ligne et donc… à être enregistré (vidéo sur youtube).

VRT est un format supporté (et inventé et maintenu) par la librairie OGR. Son rôle est de définir une donnée virtuelle, grâce à un fichier XML définissant des correspondances et traitements, et bien sûr une donnée source. On parlera de donnée VRT en parlant du fichier `.vrt`.

Exemple de contenu de fichier VRT :

```
<?xml version="1.0" encoding="UTF-8"?>
<OGRVRTDataSource>
  <OGRVRTLayer name="locations">
    <SrcDataSource relativeToVRT="1" shared="1">locations.csv</SrcDataSource>
    <SrcLayer>locations</SrcLayer>
    <Field name="city" type="String" src="CITY"/>
    <Field name="number" type="Integer" src="NUMBER"/>
    <Field name="year" type="Integer" src="YEAR"/>
    <GeometryType>wkbPoint</GeometryType>
    <LayerSRS>WGS84</LayerSRS>
    <GeometryField encoding="PointFromColumns" x="LON" y="LAT"/>
  </OGRVRTLayer>
</OGRVRTDataSource>
```

Comme on le voit dans cet exemple, le format VRT permet non seulement de définir le type de chaque champ, mais aussi de le renommer, utiliser des colonnes pour générer un champ géométrique, etc.

Je n'en dirai pas plus. Les patients attendront le cours sur le format VRT. Les curieux iront voir la doc fournie par OGR.

OGR

La librairie OGR permet de manipuler des données vectorielles, et notamment des données VRT. Elle fournit aussi des utilitaires en ligne de commande, notamment :

- ogrinfo : afficher les propriétés d'une donnée vectorielle.
- ogr2ogr : conversion d'un format à un autre. Mais en fait, bien plus que ça. Avec ses nombreuses options, et si on tient compte que parmi les formats de sortie possibles, il y a les bases de données, c'est un excellent outil de publication en base de données.

VRT + ogr2ogr = ETL ?

Combinés ensemble, VRT et ogr2ogr constituent en fait quasiment un ETL (Extract Transform Load) léger permettant de lire, transformer, filtrer et publier des données. Le tout en ligne de commande, donc facilement automatisable dans un environnement serveur. Et… gratuit et libre d'utilisation !

Le tableau comme outil de saisie de base de données

Quel rapport avec notre sujet ?

Hé bien VRT est capable de lire et transformer des données CSV. Et même du XSLX ou ODS. ogr2ogr sait publier en base de données. On a donc tout le nécessaire ou presque pour alimenter une base de données solide (type PostgreSQL, MySQL ou Oracle) en ligne, à partir de fichiers tableur gérés par des personnes non-techniques, non formées aux bases de données.

En d'autres termes : on peut

- laisser les collègues travailler sur leurs outils habituels,
  - sans leur demander d'apprendre les bases de données, la ligne de commande, les SIG etc.
  - avec peut-être juste quelques préconisations sur la façon d'organiser leurs données dans leurs tableurs
- leur permettre ainsi d'alimenter nos bases de données à vocation SIG (ou pas d'ailleurs)
- et nous, on se focalise sur notre part :
  - mettre en place le flux de publication (écrire le VRT, éventuellement automatiser la publication)
  - exploiter les données une fois en base, notamment via des vues, jointures et autre pratiques amusantes.

Bases de données

Je ne vais pas m'étaler sur la théorie, les modèles conceptuels de données etc. Vous aurez des cours sur les bases de données durant le cursus.

On va regarder la base de données dans la continuité de ce cours : un système de gestion de bases de données (SGBD), PostgreSQL par exemple, fait à peu près la même chose qu'un tableur, avec notamment les différences suivantes :

- beaucoup plus performant. Il est capable de gérer plusieurs millions d'enregistrements sans broncher
- usage distribué : habituellement, on les héberge sur un serveur, en ligne, pour des accès multiples
- sécurisé : du fait du point précédent, la sécurité importe beaucoup
- relationnel : souvent, les tables sont reliées entre elles, avec des relations de dépendances, de contraintes d'unicité etc.
- langage SQL : les interactions se font via un langage puissant et standardisé, le SQL (enfin, pas toujours, il existe des BD qualifiées de "noSQL")

Pour cette initiation, on va regarder du côté de PostgreSQL/PostGIS. PostGIS est l'extension spatiale qui permet à PostgreSQL de gérer efficacement les données à caractère géospatial.

Il existe des outils graphiques pour exploiter une base PostgreSQL. Mais pour cette mise en jambes, on est déjà sur la console alors restons-y. Il est également important de savoir se passer de l'interface graphique : vous n'y aurez pas toujours accès !

Se connecter à une base de données

1. installons le client ligne de commande pour postgresql : `sudo apt install postgresql-client`
2. il faut récupérer les paramètres de connexion de la base cible :

- hôte (nom ou IP de la machine cible)
- port (par défaut c'est 5432)

- nom de la base
- nom d'utilisateur
- mot de passe En principe, idgeo nous fournit une base, sinon j'en mettrai une en place pour la durée de ce cours

```
# note pour moi-même : si pas de base, en lancer une via la compo docker
docker compose -p idgeo-db -f resources/docker-compose.yml up -d
```

3. on utilise le client, psql, pour se connecter :

```
psql -h my_host -p 5432 -U my_user -d my_base
```

Si la connexion se passe bien, on aura une invite de commande. On peut demander l'aide via `\?`. Et il est facile de trouver une liste des commandes les plus utiles. Par exemple sur https://www.postgresqltutorial.com/postgresql-administration/psql-commands/.

Pousser de la donnée en base

A priori, notre base est vide pour l'instant. On ne va pas rentrer trop dans le détail de l'ingestion de données. Mais puisqu'on a parlé d'un flux de publication, on va voir le cas simple où l'on publie des données CSV ou xlsx en base. On va utiliser ogr2ogr :

TODO: Collecter les jeux de données à utiliser et les préparer

```
# Afin d'éviter devoir taper le mot de passe à chaque fois, on va le déclarer en variable d'enviro
export PGPASSWORD=secret  # Remplacer par le vrai mdp bien sûr
# Et pour simplifier les paramètres de connexion, on peut faire pareil avec le reste :
export PGDATABASE=cpgeom
export PGUSER=cpgeom
export PGHOST=localhost

prefix=jpommier
source=ouvrages-acquis-par-les-mediatheques
# On remplace le préfixe par défaut dans les VRT. Il permet qu'on ait chacun nos tables
# dans la base sanss'écraser les uns les autres
sed -i "s/jpommier/$prefix/g" ${source}.vrt
ogr2ogr -progress -f PostgreSQL PG:"host='$PGHOST' user='$PGUSER' dbname='$PGDATABASE'" ${source}.
# Et si ça s'est bien passé, on envoie le reste avec une boucle, c'est plus efficace :
for source in 2012-2022-balances-des-comptes-de-letat donnees-essentielles-mel-marches-publics log
do
  sed -i "s/jpommier/$prefix/g" ${source}.vrt
  ogr2ogr -progress -f PostgreSQL PG:"host='$PGHOST' user='$PGUSER' dbname='$PGDATABASE'" ${source
done
```

Pour alléger la ligne de commande, on a utilisé des variables d'environnement pour déclarer les paramètres de connexion. Je ne les ai pas nommées au hasard. Pour que psql et ogr les reconnaissent, il faut suivre une convention. Cf. https://docs.postgresql.fr/15/libpq-envars.html.

On aurait aussi pu écrire un fichier .pgpass comme documenté dans https://docs.postgresql.fr/10/libpq-pgpass.html.

Requêtes SQL (ou comment faire du tableau croisé dynamique sans souris – et sans soucis)

Commençons par vérifier que les données sont bien présentes dans la base : on va utiliser psql pour exécuter une seule commande :

```
psql -c "\dt public.*;"
```

Ca devrait nous lister les tables publiées dans le schema public, qui est le schema par défaut.

Si c'est bon, on peut ouvrir une console psql pour faire nos manips, ça sera plus pratique :

```
psql # normalement pas besoin de plus car on a défini les variables d'environnement PGTRUC préalab
```

A partir de maintenant, on est dans la ligne de commande interactive de psql

TODO:

Insertion de données

TODO: