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