

Culture numérique – module 3

Format de fichiers



Qu'est-ce qu'un format ?

- Tout fichier est un fichier **binaire** :
 - Composé de 0 et de 1.
 - Ces 0 et ces 1 peuvent être lu de **différentes façons**, comme nous l'avons vu pour les **images** et pour les **algorithmes de compressions**.
 - On parle de **format de fichier** pour designer la méthode qui a été choisie pour coder ces 0 et 1.

Pourquoi la maitrise des formats est importantes ?

- Régulièrement vos enseignants vous demandent des documents (rapports, un exercice de maths, un projet informatique...) !
- Il est parfois difficile de savoir sous quel format rendre ce qui est demandé, sous quel format travailler etc...
- Exemples d'erreurs à ne pas faire :
 - Capture d'écran de son code avant de l'envoyer à l'enseignant
 - Rapport en pdf alors qu'il doit être annoté par l'enseignant
 - Fichier dont l'extension a juste été modifiée en .zip avant envoi
 - Rapport à simplement lire, final, envoyé en word.
 - Etc...

Formats et extensions

- Quels types de fichiers utilisez-vous régulièrement ?
- Pour quels usages ?



Formats et extensions

- Différents usages possibles :
 - Image, son, vidéo
 - Bureautique
 - (Texte brut)
 - Applications, Archives
 - ...

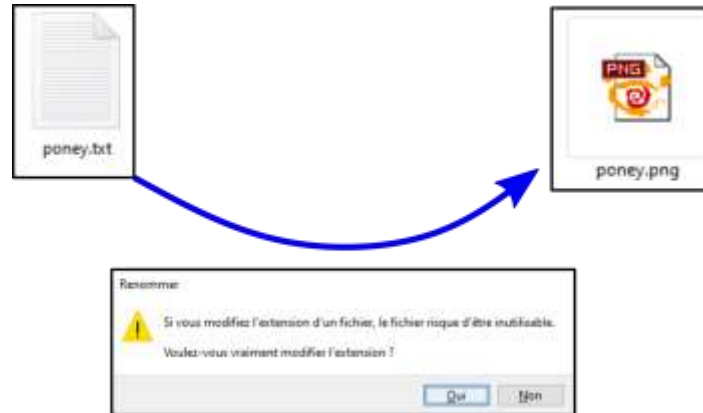


Formats et extensions

- Plusieurs formats différents pour un même usage :
 - Exemple de formats d'image ?
 - JPEG, GIF, PNG, BMP, ...
 - Différences entre ces formats :
 - Façon d'enregistrer l'information
 - Format binaire ou textuel
 - Compression (avec ou sans perte)
 - Etc.
- Comment connaître le **format** d'un fichier ?
 - C'est l'**extension** du fichier (« poney.jpg ») qui donne (*à priori*) une indication
=> Applications souvent associées à cette extension pour plus de simplicité

Formats et extensions

- L'extension d'un fichier peut tout à fait ne **PAS** correspondre à son format !
- Exemple : je renomme un fichier **texte** en **.png**



Formats et extensions

- L'extension d'un fichier peut tout à fait ne **PAS** correspondre à son format !
- Exemple : je renomme un fichier **texte** en **.png**
- Le système va tenter de l'ouvrir comme une image
 - Mais le fichier ne s'est **PAS** transformé en une image : il y aura une erreur !
 - Seul **le nom du fichier à changé**.
 - Avec de l'expérience on peut déterminer le format d'origine d'un fichier.

Formats et extensions

- Remarque :
 - On peut d'ores et déjà remarquer que les systèmes emploient un mécanisme **assez rustique** !
 - Seul le **nom du fichier** (l'extension, les dernières lettres du nom) lui donne une **indication**.
 - Pas de **mécanisme intelligent** pour détecter si les 0 et 1 du fichier sont encodés d'une manière ou d'une autre.
 - Certains logiciels tiers ou astuces pourront nous aider à pallier cette faiblesse.

Extension des fichiers, exemples :

Image	Non compressé : Bitmap : bmp	Compressé avec perte : jpg/jpeg	Compressé sans perte : gif, png
Son	Non compressé : wav	Compressé : mp3 / ogg	Compressé sans perte : FLAC
Bureautique	Office : Word : doc/docx Excell : xls/xlsx Powerpoint : pptx (des logiciels propriétaires de Microsoft)	LibreOffice : Odt (Open Document Text) Ods (Open Document Spreadsheet) Odp (Open Document Presentation) (open source)	Affichage : pdf Document alternatif : tex
Texte	Texte pur : txt, md	Données : csv, xml	Page web : html/htm (+ css / js)
Exécutables Archives	Windows : ".exe" (exécutables), ".bat" (scripts batch) ".jar" (utilisés pour les applications Java)	Linux : ".sh" (scripts shell) , deb/rmp	Archives : zip, 7z tar.gz, tar.bz

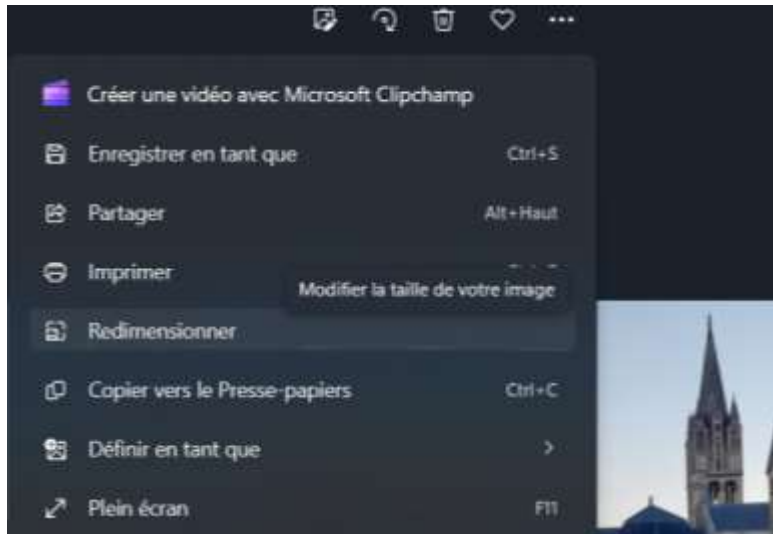
Fichiers multimédia : compléments

Format d'une image : complément

- Images = grille de pixels :
 - => plus il y a de pixels, plus le fichier est volumineux !!!
- Résolution = Nombre de pixels affichés sur une distance donnée (ppp)
 - Plus la résolution est grande, plus l'image est affichée avec des détails
 - Limite de l'écran ou de l'imprimante : Chaque dispositif d'affichage ou d'impression a sa propre limite de résolution.
 - le matériel ne peut pas afficher plus de pixels qu'il n'en est capable.
- Nécessité d'adapter la taille de l'image au format d'affichage ou d'impression et de résolution souhaité.

Redimensionner une image

- Utilisation d'applications : Photos, paint, ... (Windows) / Aperçu (Mac) / Xnview
- Photos :



Outils pratique

- Pour les vidéos et sons : **MediaInfo** permet d'obtenir le maximum d'information sur le format du fichier.
- Pour les images, **XNviewMP** permet d'afficher le maximum de donnée sur le format interne de l'image.
- Toujours pour les images : la commande identify de « **ImageMagick** » est également très efficace. (**ImageMagick** est une suite d'outils en ligne de commande pour la manipulation d'images)

Fichiers textes

Définitions

- Il s'agit de tous les fichiers qui peuvent être créés via un éditeur de texte (le bloc-notes par exemple) ou un éditeur de code.
- Les fichiers de code (.py, .c, .java, .js, .html, .php, .tex ...) sont des fichiers textes.
- Les fichiers de textes enrichis (rtf) sont des fichiers textes auxquels on ajoute des balises indiquant les styles ou la taille des polices par exemple.

Format

- Comme tous les fichiers, les données sont enregistrées sous forme **d'octets**.
- Le format standard le plus ancien encore utilisé : l'**ASCII** (American Standard Code for Information Interchange).

- **L'ASCII** (American Standard Code for Information Interchange) est un ancien standard de codage de caractères.
- Il s'agit d'un système de codage de caractères basé sur des chiffres binaires (0 et 1) qui attribue un nombre unique à chaque caractère, ce qui permet aux ordinateurs de représenter et de traiter du texte.
- 7 bits pour encoder 128 caractères : L'ASCII utilise 7 bits (séquences binaires de 0 et 1) pour représenter chaque caractère. Avec 7 bits, il est possible de représenter 2^7 , soit 128 valeurs différentes. Par conséquent, l'ASCII définit un ensemble de 128 caractères différents, allant des caractères alphanumériques aux caractères spéciaux et de contrôle.
- Majuscules, minuscules, chiffres, caractères spéciaux, espaces : L'ensemble de caractères ASCII comprend les lettres majuscules (A-Z), les lettres minuscules (a-z), les chiffres (0-9) et divers caractères spéciaux tels que les signes de ponctuation, les symboles mathématiques, les caractères de contrôle (comme la tabulation), et même un espace.

ASCII TABLE

Decimal	Hexadecimal	Binary	Octal	Char	Decimal	Hexadecimal	Binary	Octal	Char	Decimal	Hexadecimal	Binary	Octal	Char
0	0	0	0	[NULL]	48	30	110000	60	0	96	60	1100000	140	`
1	1	1	1	[START OF HEADING]	49	31	110001	61	1	97	61	1100001	141	a
2	2	10	2	[START OF TEXT]	50	32	110010	62	2	98	62	1100010	142	b
3	3	11	3	[END OF TEXT]	51	33	110011	63	3	99	63	1100011	143	c
4	4	100	4	[END OF TRANSMISSION]	52	34	110100	64	4	100	64	1100100	144	d
5	5	101	5	[ENQUIRY]	53	35	110101	65	5	101	65	1100101	145	e
6	6	110	6	[ACKNOWLEDGE]	54	36	110110	66	6	102	66	1100110	146	f
7	7	111	7	[BELL]	55	37	110111	67	7	103	67	1100111	147	g
8	8	1000	10	[BACKSPACE]	56	38	111000	70	8	104	68	1101000	150	h
9	9	1001	11	[HORIZONTAL TAB]	57	39	111001	71	9	105	69	1101001	151	i
10	A	1010	12	[LINE FEED]	58	3A	111010	72	:	106	6A	1101010	152	j
11	B	1011	13	[VERTICAL TAB]	59	3B	111011	73	;	107	6B	1101011	153	k
12	C	1100	14	[FORM FEED]	60	3C	111100	74	<	108	6C	1101100	154	l
13	D	1101	15	[CARRIAGE RETURN]	61	3D	111101	75	=	109	6D	1101101	155	m
14	E	1110	16	[SHIFT OUT]	62	3E	111110	76	>	110	6E	1101110	156	n
15	F	1111	17	[SHIFT IN]	63	3F	111111	77	?	111	6F	1101111	157	o
16	10	10000	20	[DATA LINK ESCAPE]	64	40	1000000	100	@	112	70	1110000	160	p
17	11	10001	21	[DEVICE CONTROL 1]	65	41	1000001	101	A	113	71	1110001	161	q
18	12	10010	22	[DEVICE CONTROL 2]	66	42	1000010	102	B	114	72	1110010	162	r
19	13	10011	23	[DEVICE CONTROL 3]	67	43	1000011	103	C	115	73	1110011	163	s
20	14	10100	24	[DEVICE CONTROL 4]	68	44	1000100	104	D	116	74	1110100	164	t
21	15	10101	25	[NEGATIVE ACKNOWLEDGE]	69	45	1000101	105	E	117	75	1110101	165	u
22	16	10110	26	[SYNCHRONOUS IDLE]	70	46	1000110	106	F	118	76	1110110	166	v
23	17	10111	27	[ENG OF TRANS. BLOCK]	71	47	1000111	107	G	119	77	1110111	167	w
24	18	11000	30	[CANCEL]	72	48	1001000	110	H	120	78	1111000	170	x
25	19	11001	31	[END OF MEDIUM]	73	49	1001001	111	I	121	79	1111001	171	y
26	1A	11010	32	[SUBSTITUTE]	74	4A	1001010	112	J	122	7A	1111010	172	z
27	1B	11011	33	[ESCAPE]	75	4B	1001011	113	K	123	7B	1111011	173	{
28	1C	11100	34	[FILE SEPARATOR]	76	4C	1001100	114	L	124	7C	1111100	174	
29	1D	11101	35	[GROUP SEPARATOR]	77	4D	1001101	115	M	125	7D	1111101	175	}
30	1E	11110	36	[RECORD SEPARATOR]	78	4E	1001110	116	N	126	7E	1111110	176	~
31	1F	11111	37	[UNIT SEPARATOR]	79	4F	1001111	117	O	127	7F	1111111	177	[DEL]
32	20	100000	40	[SPACE]	80	50	1010000	120	P					
33	21	100001	41	!	81	51	1010001	121	Q					
34	22	100010	42	"	82	52	1010010	122	R					
35	23	100011	43	#	83	53	1010011	123	S					
36	24	100100	44	\$	84	54	1010100	124	T					
37	25	100101	45	%	85	55	1010101	125	U					
38	26	100110	46	&	86	56	1010110	126	V					
39	27	100111	47	'	87	57	1010111	127	W					
40	28	101000	50	(	88	58	1011000	130	X					
41	29	101001	51	)	89	59	1011001	131	Y					
42	2A	101010	52	*	90	5A	1011010	132	Z					
43	2B	101011	53	+	91	5B	1011011	133	[					
44	2C	101100	54	,	92	5C	1011100	134	\					
45	2D	101101	55	-	93	5D	1011101	135	]					
46	2E	101110	56	.	94	5E	1011110	136	^					
47	2F	101111	57	/	95	5F	1011111	137	_					

- Pour coder et décoder des caractères, on se réfère à une table ASCII, qui associe chaque caractère à sa valeur binaire ou hexadécimale. Voici un exemple simplifié de quelques caractères en ASCII :
 - Caractère "A" : En binaire, c'est "01000001". En hexadécimal, c'est "41".
 - Caractère "B" : En binaire, c'est "01000010". En hexadécimal, c'est "42".
 - Caractère "C" : En binaire, c'est "01000011". En hexadécimal, c'est "43".

Extension du format

- Formats **construits sur l'ASCII** : les formats ISO 8859 (en France : 8859-1 ou latin-1 / 8859-15) ou cp1252 (Windows)

Profite du fait que les ordinateurs utilisent majoritairement l'octet (**8 bits**) et qu'il y a donc un bit **inutilisé** avec l'ASCII !

-> les 128 caractères restant permettent d'ajouter des **caractères supplémentaires** : les caractères accentués, les caractères € ou œ

-> En résumé, les formats ISO 8859 et Windows-1252 sont des extensions de l'ASCII qui permettent de représenter un ensemble plus large de caractères en utilisant un octet complet (8 bits)

Limites et solutions ...

- Limite des extensions du format ASCII :
 - Tous les alphabets ne sont pas concernés par un même format
 - Les différentes langues, systèmes, OS, vont utiliser des versions différentes
 - N'oublions pas que les formats sont difficiles à reconnaître par le système
 - Conséquence ?

Fonction PHP du jour : levenshtein()

Posted by Blount on 21 octobre 2010 [Edit]

Je vais tenter d'écire périodiquement un billet afin de traiter d'une fonction PHP indispensable, ou encore qui me semble simplement utile à connaître.

Pour aujourd'hui, la fonction sera levenshtein().

Présentation

Du nom de son auteur, Vladimir Levenshtein, la distance Levenshtein permet, en g nombre représentant la distance entre deux chaînes de caractères.

Pour cela, l'algorithme détermine un coût basé sur le nombre de substitutions, suppression de caractères afin de transformer le premier mot vers le second. Plus présenteront des différences, plus le nombre sera élevé.

Pour plus d'informations, je vous redirige sur la page de Wikipedia, d'où je me s

» Lire la suite!

Limites et solutions ...

- Comment résoudre le problème ?
 - En attribuant plus de bits au codage d'un caractère
- Mais on aimerait que les anciens textes codé en ASCII soient toujours compatible ! N'oublions pas que l'ordinateur est mauvais pour deviner un format !

Limites et solutions ...

- Exemple du problème :
 - 'A' se code **01000001** en **ASCII**
 - Si je veux plus de place je dis que mon code compte 2 octets, par exemple
 - 'A' se code donc **01000001 00000000**

- Sauf qu'un texte sous ce format signifiant « AA » codera donc A0A0 si on le lit en ASCII !
 - **Incompatible !**
 - Comment résoudre le problème ?

Limites et solutions ...

-> Solution proposée par l'UTF-8 (Unicode Transformation Format - 8 bits)

Intérêt : Nombre d'octet variable !

1 octet, codant jusqu'à 7 bits	0 bbbb·bbbb
2 octets, codant jusqu'à 11 bits	110 b·bbbb 10 bb·bbbb
3 octets, codant jusqu'à 16 bits	1110· bbbb 10 bb·bbbb 10 bb·bbbb
4 octets, codant jusqu'à 21 bits	1111·00 bb 10 bb·bbbb 10 bb·bbbb 10 bb·bbbb
	1111·0100 1000· bbbb 10 bb·bbbb 10 bb·bbbb

- **Caractères ASCII** : Les caractères ASCII standard, tels que les lettres de l'alphabet anglais, les chiffres et les symboles de base, sont représentés sur un seul octet, identique à leur représentation ASCII standard. Par exemple, la lettre "A" est toujours codée comme 0x41 (65 en décimal) en UTF-8.
- **Caractères étendus** : Les caractères qui ne sont pas dans l'ensemble de caractères ASCII standard sont représentés sur plusieurs octets en UTF-8. Par exemple, le caractère "é" (e accent aigu) est codé en UTF-8 sur deux octets : 0xC3 0xA9.
- **Caractère Unicode** : L'UTF-8 peut également représenter des caractères Unicode qui nécessitent plus de deux octets.

- **Exemple:**

Texte en UTF-8 : Café

Représentation binaire :

- "C" : 0x43 (caractère ASCII)
- "a" : 0x61 (caractère ASCII)
- "f" : 0x66 (caractère ASCII)
- "é" : 0xC3 0xA9 (caractère étendu)
- " " : 0x20 (caractère ASCII espace)

En UTF-8, ce texte serait stocké sous forme binaire comme suit :

01000011 01100001 01100110 11000011 10101001 00100000

Limites et solutions ...

-> Inconvénient de la solution ?

On doit rester 100% compatible ASCII, mais le ASCII a quelques spécificités :

Exemple : Les caractères de « fin de chaine », qui définissent la fin d'une ligne, sont différemment codé selon les systèmes !

Un texte ASCII de plusieurs lignes sous Windows n'est pas le même que sous Linux ou MAC !

-> IDEM sous UTF-8 malheureusement ...

Résumé

ASCII

0XXXXXXX

7 bits

ISO 8859 ou cp1252

XXXXXXXX

8 bits

UTF-8

0XXXXXXX

7 bits

Caractères de l'ASCII

110XXXXX

10XXXXXX

11 bits

Exemple : accents, monnaies, alphabet Grec, Arabe...

1110XXXX

10XXXXXX

10XXXXXX

16 bits

Exemple : caractères Chinois, Japonais, Braille ...

11110XXX

10XXXXXX

10XXXXXX

10XXXXXX

21 bits

Exemple : Hiéroglyphes...

Quelques conseils

- Qu'apprendre de ces différents types d'encodages ?
- > Quand vous **codez**, n'utilisez que des caractères reconnus par l'ASCII, sinon :
 - Si vous partagez votre code avec quelqu'un, vous devrez vous assurer d'utiliser le même encodage
 - Si vous ouvrez votre code avec un éditeur différent, vérifiez que l'encodage est identique
 - Problèmes classiques :
 - cp1252 et ISO 8859 n'encodent pas les retours de lignes de la même façon
 - Un caractère accentué en UTF-8 sera remplacé par deux caractères incohérents dans un des autres encodage (exemple : « é » donnera « Ã© » en cp1252)

Quelques conseils

- Logiciel pratique quand on ne sait pas de quel encodage est un fichier :

- Notepad ++



- Contrairement aux systèmes d'exploitation :
 - Peut ouvrir et encoder dans la plupart des types d'encodages
 - Mais surtout : **utilise des algorithmes malins pour détecter le format réel d'un fichier texte**

Les formats de textes « formatés »

Kesako ?

- Pour aller au delà des fichiers textes on a parfois besoin :
 - De rajouter de la mise en forme (**gras**, souligné, **couleurs** etc.)
 - D'organiser le texte (bordure d'une page, en tête et pied de page etc...), centrage du texte etc.
 - De rajouter d'autres éléments (images etc...)
 - De mettre en forme sous forme de tableau et de rajouter des formules de calcul automatique (pour tenir une comptabilité etc.)
- De nombreux logiciels existent pour **éditer** et ensuite **afficher** tout ceci.
 - Les données sont alors sauvegarder sous forme d'un fichier au format bien spécifique !

Quelques exemples répondant à cette problématique ...

Exemple simple, le RTF :

Si je tape le texte ci-dessous :

```
\rtf1\ansi
\fonttbl\font\fswiss Arial;
\font\pard Voici du texte en
\b bold
  gras.\par
```



Je sauvegarde et je renomme
mon fichier texte en **toto.rtf**

J'obtiens :

« Voici du texte en **gras** »

Exemple simple, le RTF :

C'est un système qu'on dit « descriptif » utilisant des « balises »

→ Vous rencontrerez cette astuce très souvent.

- La plupart des langages du web s'inspirent de ce type de fonctionnement
- Le langage de description le plus connu dans le domaine scientifique est très proche (Tex)

Un format texte formateur, le .md

- Format texte utilisé pour le code
- Système de « balises » simplifié (comme html, mais en plus simple)
- Très utilisé par plusieurs systèmes de documentation de code (ex. gitlab, github...)
- Titres : Les titres sont généralement indiqués en utilisant un certain nombre de dièses (#) suivies d'un espace, suivi du texte du titre.

Par exemple :

Titre principal

Titre niveau 2

Titre niveau 3

Un format texte formateur, le .md

Formatage de texte et de balisage sémantique: la rédaction de documentation, de fichiers README, de courriels et d'autres types de documents en texte brut. Voici comment chaque élément de formatage fonctionne :

- **Italique** : Pour mettre du texte en italique, vous pouvez l'entourer d'astérisques (*) ou d'underscores (_).

Par exemple :

Italique ou *_italique_*

- **Gras** : Pour mettre du texte en gras, vous pouvez l'entourer de doubles astérisques (**) ou de doubles underscores (__).

Par exemple :

****Gras**** ou **__gras__**

*****Gras et italique***** ou **__gras et italique__**

- **Liens hypertextes** : Pour créer un lien hypertexte, vous pouvez utiliser la syntaxe [texte du lien](http://url.du.lien).

Par exemple :

[texte de lien](<http://url.du.lien>)

- **Ajout de code préformaté** : Pour inclure du code préformaté, vous pouvez utiliser des triple backticks (``), suivi de la langue de programmation (facultative), puis du code sur plusieurs lignes, et enfin, terminez par des triple backticks.

Par exemple :

```
```
def fonction():
print("Ceci est du code")
```
```

=> **Résultat** :

```
Def fonction():
    print("Ceci est du code")
```

Un format texte formateur, le .md

- **Citation :**
> Ce texte apparaîtra dans un élément HTML <blockquote>.
- **Lists:**
 - non numérotées (marche aussi avec *)
 - foo
 - bar
 - 1. numérotées
 - 2. baz
- **Possibilité d'imbriquer avec tab**
 - liste 1
 - liste 2, item 1
 - liste 2, item 2

Un format texte formateur, le .md

- **Tableaux** (pénible, utiliser [ce site](#) par exemple)
- Exemple: Pour créer des tableaux en Markdown, vous pouvez utiliser la syntaxe suivante :

| En-tête 1 | En-tête 2 | En-tête 3 |

|-----|-----|-----|

| Ligne 1, Colonne 1 | Ligne 1, Colonne 2 | Ligne 1, Colonne 3 |

| Ligne 2, Colonne 1 | Ligne 2, Colonne 2 | Ligne 2, Colonne 3 |

- Qui donnera un tableau comme suit:

En-tête 1	En-tête 2	En-tête 3
Ligne 1, Colonne 1	Ligne 1, Colonne 2	Ligne 1, Colonne 3
Ligne 2, Colonne 1	Ligne 2, Colonne 2	Ligne 2, Colonne 3



Images:

! [description alternative] (url/ou/chemin/de/l'image)

Exemple:

! [Logo Markdown] (<https://example.com/logo.png>)

- Plusieurs extensions possibles (pour les maths par exemple)
 - Pour visualiser le résultat en Markdown, vous pouvez utiliser les éditeurs suivants :
 1. Atom : Vous pouvez utiliser Atom avec des extensions Markdown comme "markdown-preview" pour prévisualiser votre Markdown et voir le rendu en HTML en temps réel.
 2. Notepad++ : Pour utiliser Notepad++, vous pouvez également ajouter des extensions Markdown telles que "MarkdownViewerPlusPlus" pour prévisualiser votre Markdown. Cette extension affichera le résultat dans une fenêtre séparée.
- > En utilisant ces extensions et éditeurs, vous pouvez facilement travailler avec des tableaux, des images et d'autres éléments Markdown tout en visualisant le rendu final

Un format texte formateur, le .md

```

* Administration Linux
**Semestre** 6 (CIR)
**UE** Informatique
**ECUE** Administration Linux
**Prérequis** Avoir Linux sur sa machine
**Objectif en termes de compétence** connaître les principales commandes, les enjeux de
l'administration et les principaux processus permettant d'administrer son propre système Linux.
** Programme

Le programme consiste dans un premier temps en un rappel sur les commandes shell de base, le
système de fichiers et les processus dans un système Linux. Le second cours est dédié au langage
'awk' et à l'utilisation des expressions régulières (ou regex) pour faire du parsing. Nous nous
intéresserons ensuite à la gestion avancée des utilisateurs, des groupes et des mots de passe,
puis au système 'cron', qui permet de gérer l'automatisation l'exécution des scripts et
commandes. Durant la même séance, il y aura un cours-TP sur les systèmes de gestion de version,
en particulier 'git'. Les deux derniers cours seront dédiés à des aspects liés au réseau : la
gestion du réseau sur un système Linux (sockets, configuration réseau etc.), et la sécurité qui
va avec, et connexion à distance à travers 'ssh'. Enfin, les 4 derniers cours seront dédiés à la
réalisation d'un projet de conception d'un rapport régulier automatique de données
météorologiques à partir de données présentes sur un serveur, l'évaluation du module sera
réalisée à travers l'évaluation de ce projet et d'un devoir sur table en particulier pour les
parties qui n'auront pas pu être évaluées durant le projet.

```

Sujet	**Cours (h)**	**TP (h)**	**TD (h)**	**Date et heure**
Rappels	1		2	25/03
8h30				
awk et regex	1		2	30/03
13h30				
Gestion des utilisateurs	1		2	01/04
8h30				
Crontab & git	1		2	05/04
13h30				
Réseau & firewall	1		2	08/04
8h30				
ssh	1		2	20/04
8h30				

Administration Linux

Semestre 6 (CIR) UE Informatique ECUE Administration Linux **Prérequis** Avoir Linux sur sa machine
Objectif en termes de compétence connaître les principales commandes, les enjeux de
l'administration et les principaux processus permettant d'administrer son propre système Linux.

Programme

Le programme consiste dans un premier temps en un rappel sur les commandes shell de base, le système de fichiers et les processus dans un système Linux. Le second cours est dédié au langage 'awk' et à l'utilisation des expressions régulières (ou regex) pour faire du parsing. Nous nous intéresserons ensuite à la gestion avancée des utilisateurs, des groupes et des mots de passe, puis au système 'cron', qui permet de gérer l'automatisation l'exécution des scripts et commandes. Durant la même séance, il y aura un cours-TP sur les systèmes de gestion de version, en particulier 'git'. Les deux derniers cours seront dédiés à des aspects liés au réseau : la gestion du réseau sur un système Linux (sockets, configuration réseau etc.), et la sécurité qui va avec, et connexion à distance à travers 'ssh'. Enfin, les 4 derniers cours seront dédiés à la réalisation d'un projet de conception d'un rapport régulier automatique de données météorologiques à partir de données présentes sur un serveur. L'évaluation du module sera réalisée à travers l'évaluation de ce projet et d'un devoir sur table en particulier pour les parties qui n'auront pas pu être évaluées durant le projet.

Sujet	Cours (h)	TP (h)	TD (h)	Date et heure
Rappels	1		2	25/03 8h30
awk et regex	1		2	30/03 13h30
Gestion des utilisateurs	1		2	01/04 8h30

Les formats « bureautique »

Pourquoi ?

- On a souvent besoin d'aller plus loin que le simple format RTF ou MD.
 - Par exemple : pour écrire un rapport.
- Pour répondre à ces besoins, des logiciels spécifiques définissent leurs propres formats, souvent assez complexe.

Exemples :

- Quelques formats de bureautique :
 Document : doc, docx, odt
 Tableurs : xls,xlsx, ods
 Présentation : ppt, pptx, odp
- Manipulables par des logiciels de bureautique spécifiques (**Libre Office**, **Microsoft Office**, Google Documents...)
- Non éditables par d'autres outils. (éditeurs de textes)
- Peu compatible entre eux...
- Outils de type « Wysiwyg » : « ***What you see is what you get*** » : **Outils de type "Wysiwyg"** : Les outils de type "Wysiwyg" (What you see is what you get) sont des logiciels qui permettent aux utilisateurs de voir le résultat final de leur travail à l'écran, tel qu'il apparaîtra à l'impression ou à l'affichage, sans nécessiter une connaissance approfondie de la programmation ou de la syntaxe.

Usages :

Partage de fichiers documents :

- Ces documents, une fois transmis, restent modifiables (certains systèmes de protection existent).
- Ces documents peuvent potentiellement s'afficher différemment selon la version ou l'outil utilisé (cf slides suivants).



TP-Commandes LINUX: 1ère approche

Commandes LINUX : 1^{ère} approche



Objectifs du TP : commandes minimales pour utiliser un ordinateur sous Linux, sans interface graphique, uniquement en ligne de commande. Donc :

- comprendre l'arborescence de fichiers et connaître les commandes relatives à cette arborescence (`mkdir`, `cd`, `ls`, `pwd`, ...).
- savoir éditer un fichier et utiliser quelques commandes de base sur les fichiers (`cat`, `cp`, `rm`, ...).
- connaître quelques commandes sur les processus (`ps`, `kill`).
- savoir installer des packages binaires et compiler ses propres programmes en binaire.

1. Qu'est ce que Linux?

Linux est un système d'exploitation libre, implémentation "open source" du système Unix.

Il est important de noter que connaître les commandes sous Linux permet de savoir utiliser tous les systèmes dits "Unix-like" (divers Unix commerciaux, OpenBSD, FreeBSD, etc.).

La part de marché de Linux, faible sur les ordinateurs personnels (~2%), est élevée sur les serveurs (~35% Linux, ~65% tous "Unix-like" confondus) et les systèmes embarqués (~30%).

A la différence des systèmes "propriétaire" Windows (Microsoft) et Mac OS X (Apple), qui sont monolithiques, le côté "libre" de Linux favorise l'existence de différentes branches ("forks"), appelées "distributions Linux". Ce sont des suites logicielles toutes basées sur ce qu'on appelle le "noyau Linux" ("Linux kernel"). Il existe des centaines de distributions, mais on peut les regrouper en quelques grandes familles dont le nom est donné par la distribution initiale (historiquement) :

- la famille  Debian, avec notamment la  distribution Ubuntu,
- la famille  Red Hat, avec les distributions  Fedora,  CentOS,  Mageia (récent),

et des familles un peu moins importantes :

-  Slackware,
-  Arch Linux,
- etc...

2. Le mode commande" mode "texte", "console")

Linux dispose d'un analogue à Windows et convivial. être nécessaire "ligne de commande" (beaucoup d'utilisateurs expérimentés de Linux préfèrent d'ailleurs utiliser ce mode), pour plusieurs raisons :

- le mode graphique est consommateur de ressources processeur et mémoire, ce qui est gênant pour des systèmes ayant des ressources limitées (carte d'un système embarqué, routeur, ...) ou pour des serveurs fortement sollicités. Pour ce type de machines, il n'y a pas de mode graphique installé.
- la connexion en ligne de commande sur des machines distantes (par `telnet` ou `ssh`) est moins consommatrice en bande passante réseau qu'une connexion en mode graphique.
- en cas de certaines pannes logicielles (ex: "freeze" du système graphique), le mode "ligne de commande" est indispensable pour s'en sortir.
- les commandes en mode "texte", grâce à leurs nombreuses options et à la possibilité de les chaîner en "pipeline", permettant une grande souplesse et sont plus polyvalentes ("versatile") qu'une application graphique.

Syntaxe générale d'une commande : commande [options] paramètres

En général, les options (souvent une seule lettre) sont précédées du symbole "-" et peuvent être groupées. Les paramètres indiquent en général le(s) fichier(s) concerné(s). Exemple de commande :

```
tar -xzvf fichier.tgz
```

avec commande : `tar`
options : `x`, `z`, `v` et `f`
paramètre : `fichier.tgz`

Attention ! bien respecter la casse des caractères (c'est-à-dire les majuscules et minuscules).

Remarque : sous Unix/Linux l'interpréteur de commandes s'appelle le "shell" (coquille).

3. Pratiquement, pour démarrer :

d'utilisation par linuxcounter.net

"ligne de (appelé aussi "terminal" ou

mode graphique, et Mac, très complet. Cependant, à peut d'utiliser le mode

Les formats imprimables

Fichiers imprimables

- Fichiers décrivant et contenant **tout** ce dont a besoin un fichier pour être correctement visualisé :
 - Caractères (polices)
 - Dessins vectoriels
 - Images Bitmap

- Un fichier PDF s'affiche **EXACTEMENT** de la même manière sur un ordinateur avec un matériel A et un logiciel B et sur un autre ordinateur avec un matériel C et un logiciel D.

Usages ?

- Lorsqu'on veut partager un document non modifiable (généralement) qui aura exactement la même présentation.
- Utile pour partager des rapports, des articles, etc etc...
- Vos rapports doivent très souvent être rendu en PDF !

Historique et particularités

- Format créé à la base par Adobe et propriétaire jusqu'en 2008
- Depuis 2008 : format libre, mais certaines fonctionnalités restent encore difficilement accessible sans logiciel payant (par exemple : l'édition de PDF, l'annotation).
- PDF (Portable Document Format): fait suite au Postscript :
 - commence toujours par un « nombre magique » suivi d'un identifiant de version
 - Exemple : %PDF-1.4

Historique et particularités

- On ne peut pas éditer directement un fichier pdf
 - Ou plutôt : assez difficilement, et uniquement avec certains logiciel spécifique comme Adobe Acrobat, mais ce n'est souvent utilisé que pour des retouches, pas pour créer le fichier d'origine !
- Pour créer un fichier PDF on peut :
 - **Exporter** un fichier au format pdf depuis une application (word par exemple)
 - **Imprimer** un fichier en utilisant le driver dédié (exemple windows : Microsoft print to pdf)
 - Le **générer** à partir d'un outil dédié le générant à partir d'un langage (TeXToPDF par exemple, génère à partir du langage Tex/Latex).
- Certains outils permettent de dessiner ou prendre des notes sur un pdf.

Les executables

- Un exécutable est un fichier informatique qui contient un programme ou une série d'instructions destinées à être exécutées par un ordinateur.
- Il entraîne la création d'un processus dans le système
- On considère parfois que seuls les fichiers interprétables par le système d'exploitation sont des exécutables (exclusion des .py, .bat, .sh)
- Exemple:

Gestionnaire des tâches

Fichier Options Affichage

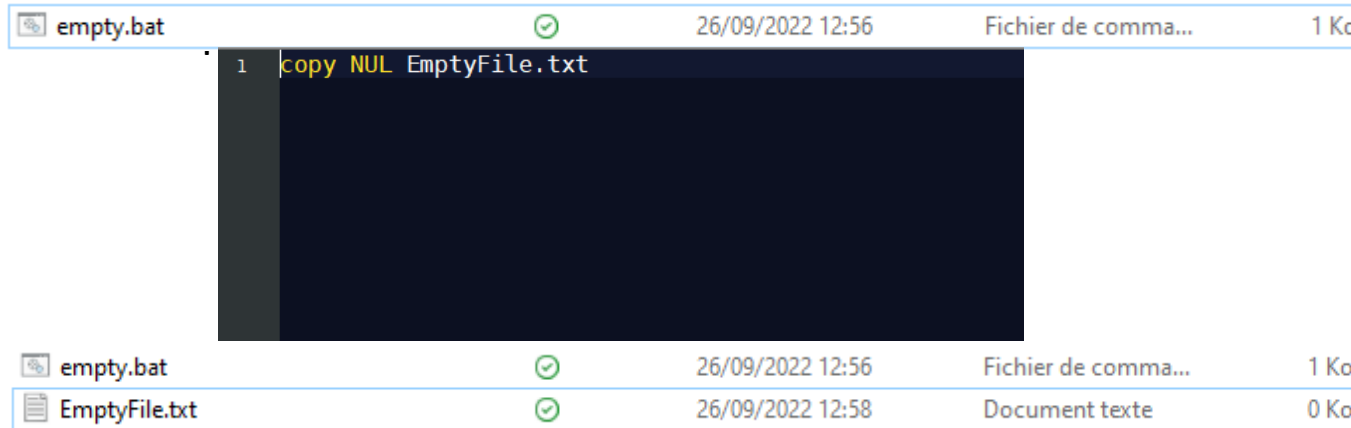
Processus Performance Historique des applications Démarrage Utilisateurs Détails Services

Nom	Statut	42%	87%	33%	0%	2%		
		Processeur	Mémoire	Disque	Réseau	Processe...	Moteur de processeur graphique	Cons
Microsoft Teams		1,0%	182,1 Mo	0 Mo/s	0 Mo/s	0%		Très
> Microsoft PowerPoint (2)		0%	174,0 Mo	0 Mo/s	0 Mo/s	0%		Très
Firefox		0%	157,2 Mo	0 Mo/s	0 Mo/s	0%		Très
Paint		0%	126,1 Mo	0 Mo/s	0 Mo/s	0%		Très



Les exécutables

- Sous Windows:
 - La majorité des exécutables sont des .exe (interprétables par l'OS)
 - Il existe aussi des .msi (fichiers d'installation)
 - Dans une acception plus large, on peut aussi compter les .bat (scripts interprétables en MS-DOS)



Les exécutables

- Sous Linux :
 - Pas d'extension spécifique
 - Le fichier a un *droit* particulier (droit d'exécution pour une classe d'utilisateurs)
 - Appimage: logiciels portables; .deb/rpm: fichiers d'installation
 - Fichiers scripts OS également (.sh)

```
-rwxrwxrwx 1 cburon cburon      20 sept. 26 13:00 empty.sh*
```

```
1 touch EmptyFile.txt
2
```

```
$ ./empty.sh ; ls -l
```

```
-rwxrwxrwx 1 cburon cburon      0 sept. 26 13:02 EmptyFile.txt
-rwxrwxrwx 1 cburon cburon     20 sept. 26 13:00 empty.sh
```

- En résumé, sous Linux, les exécutable peuvent avoir diverses extensions ou aucune extension du tout. Les droits d'exécution sont essentiels pour permettre l'exécution d'un fichier. Les formats de distribution de logiciels tels qu'AppImage, .deb et .rpm, ainsi que les fichiers scripts (.sh), sont couramment utilisés pour distribuer et exécuter des logiciels sous Linux.

Les exécutables

- Sous Mac:
 - Les .sh fonctionnent aussi ! (base unix)
 - Autre format: les .dmg



Quelques notes sur les executables

- Le **code** est au **format texte**
- Selon l'extension, on détermine le langage...
- Mais attention ! Cette extension ne change rien pour le fichier ni pour l'ordinateur ! Il n'est là qu'à titre indicatif
- On peut exécuter/compiler n'importe quel fichier source avec n'importe quel compilateur/interpréteur...À condition qu'il soit bien écrit.
- Une fois compilé, le fichier se transforme en format binaire (un .exe par exemple)
- Au contraire : si on est sur un langage interprété, alors l'interpreteur va lire notre code (exemple python)
- Des cas intermédiaires existes (Java : compilé en langage intermédiaire qui est ensuite lu par la JVM)

Fichier local vs distant

- 1 ressource/fichier local :
 - + Toujours dispo, pas de connexion réseau nécessaire pour le consulter, accès plus rapide
 - Pas de sauvegarde automatique du fichier, nécessite de l'espace suffisant sur le disque dur
- 1 ressource/fichier distant :
 - + Sauvegarde automatique, utilisable par plusieurs périphériques ou utilisateurs en même temps, n'encombre pas le disque dur
 - Nécessite une connexion réseau
- 1 ressource/fichier à la fois local et distant :
 - + Sauvegarde automatique, disponible sans connexion
 - Ressources réseaux nécessaires à chaque modification du fichier
- 1 ressource partagée :
 - + Modifications simultanées d'un même fichier
- Quels sont les avantages et inconvénients de chaque cas d'usage ?

Fichier créé localement / Fichier système

Stream video Youtube, Netflix, Spotify, ...

Répertoire Dropbox, OneDrive, git

Document GoogleDocs, Teams, Overleaf

Conversion de fichiers

Conversion de fichiers : liste d'outils

- Doc -> PDF
 - Outils de bureautique, impression vers PDF
- Txt -> doc
 - Word peut l'ouvrir mais demande un encodage : Copier / coller sinon
- Doc -> txt
 - Copier/coller uniquement (seul le texte est copié)
- PDF -> Doc ou txt
 - Impossible, et les copier/coller ne fonctionnent pas toujours (pb encodage)
- N'importe quel fichier(s) -> zip
 - Outil de compression du système, 7zip, Winzip ...
- zip -> N'importe quel fichier(s)
 - Outil de compression du système, 7zip, Winzip ...
 - Sauf que là... on décompresse

Conversion de fichiers: explications

- **Doc -> PDF** : Vous pouvez convertir un document (par exemple, au format Microsoft Word) en un fichier PDF en utilisant des outils de bureautique ou en imprimant le document en tant que fichier PDF. Cela permet de créer une version statique et portable du document.
- **Txt -> Doc** : Les fichiers texte peuvent être convertis en documents (par exemple, au format Microsoft Word) en copiant et collant le contenu dans un document Word. Cependant, cela peut nécessiter un certain ajustement de la mise en forme, car le texte brut n'inclut généralement pas de formatage complexe.
- **Doc -> Txt** : Vous pouvez extraire le texte brut d'un document (par exemple, au format Microsoft Word) en le copiant et en le collant dans un fichier texte. Seul le texte est copié, sans mise en forme ni éléments graphiques.

- **PDF -> Doc ou Txt** : La conversion de fichiers PDF en documents Word ou fichiers texte peut être difficile, car les PDF peuvent contenir des éléments complexes et la conversion peut ne pas toujours préserver correctement la mise en forme. Les copier/coller à partir d'un PDF peut également entraîner des problèmes d'encodage.
- **N'importe quel fichier(s) -> Zip** : Vous pouvez compresser un ou plusieurs fichiers (quelle que soit leur nature) en utilisant un outil de compression tel que 7-Zip ou WinZip. Cela permet de regrouper plusieurs fichiers en un seul fichier ZIP pour économiser de l'espace de stockage ou faciliter la distribution.
- **Zip -> N'importe quel fichier(s)** : Vous pouvez extraire le contenu d'un fichier ZIP (qui peut contenir n'importe quel type de fichier) en utilisant un outil de décompression comme 7-Zip ou WinZip. Cela vous permet de récupérer les fichiers originaux à partir de l'archive ZIP.



Conversion de fichiers : l'outil ultime pandoc

- Outil de conversion multi-formats très riche, trop de format pour tous les afficher !
- Quelques exemples :
 - Entrées : docx, epub, html, latex, markdown, odt, rst, rtf
 - Sorties : docx, epub, html, **pptx**, latex, markdown, odt, rst, rtf, **pdf**
- De *très* nombreuses options disponibles selon les formats !
- Outil Open Source, [téléchargeable](#) sous Windows comme sous Linux ou MacIntosh
- Il existe aussi [une version d'essai en ligne](#)

