

# Apprendre les bases d'Asciidoc

v1.0.0 | 23/08/2024 à 21:08 | Auteur : Emmanuel Ravrat

Livre

**Support généré avec  
le framework**

**A s c i i d o c P r o**

# Table des matières

<b>1. Préparation de l'IDE pour éditer du contenu AsciiDoc</b>	<b>1</b>
1.1. Création d'un premier document Asciidoc dans l'IDE	1
1.2. Installation du plugin Asciidoc	6
1.3. Générer un fichier pdf à partir du fichier Asciidoc	8
<b>2. Gestion de la langue française dans l'IDE</b>	<b>10</b>
2.1. Installation d'un outil de vérification du texte	10
2.2. Configuration de la langue à inspecter	11
2.3. Ajout de dictionnaires	11
<b>3. Utiliser son navigateur pour prévisualiser un fichier AsciiDoc</b>	<b>14</b>
3.1. Installation du plugin Asciidoctor dans son navigateur	14
3.2. Les options du plugin de rendu Asciidoc	15
<b>4. Les commentaires</b>	<b>18</b>
<b>5. Les titres</b>	<b>20</b>
5.1. L'importance des titres	20
5.2. Le titre principal d'un document AsciiDoc	20
5.3. La structure du document	21
5.4. La table des matières	24
5.4.1. Afficher la table des matières	24
5.4.2. Personnaliser la table des matières	26
<b>6. Les paragraphes</b>	<b>29</b>
<b>7. Mise en forme du texte</b>	<b>33</b>
<b>8. Mise en forme avec les rôles intégrés</b>	<b>35</b>
<b>9. Les listes</b>	<b>38</b>
9.1. Créer des listes non ordonnées	38
9.2. Créer des listes ordonnées	41
9.3. Créer des checklists	45
<b>10. Les images</b>	<b>46</b>
10.1. Copier-coller une image	46
10.2. Insertion et fonctionnalités concernant les images	46
10.3. Positionner une image	50
<b>11. Les liens</b>	<b>52</b>
11.1. Les liens web	52
11.2. Les liens vers des fichiers	53
<b>12. Utiliser des icônes</b>	<b>56</b>
<b>13. Les références croisées</b>	<b>59</b>
13.1. Qu'est-ce qu'une référence croisée ?	59
13.2. Référencer un titre	60
13.3. Référencer un bloc : paragraphe, tableau, liste, etc.	63

13.4. Référencer un élément en "ligne" : des mots, un item de liste, etc. . . . .	64
13.5. Voir où une ancre est utilisée et comment la renommer . . . . .	66
<b>14. Les blocs d'avertissement (admonitions) . . . . .</b>	<b>68</b>
14.1. Qu'est-ce qu'un bloc d'avertissement ? . . . . .	68
14.2. Ecrire des blocs d'avertissement . . . . .	68
<b>15. Les blocs de code source . . . . .</b>	<b>73</b>
15.1. Le bloc de liste . . . . .	73
15.2. Du code dans un bloc de liste . . . . .	73
15.3. Autres fonctionnalités liées aux blocs de code . . . . .	74
15.3.1. Afficher les numéros de lignes . . . . .	75
15.3.2. Mettre en surbrillance une ou plusieurs lignes . . . . .	75
15.3.3. Contrôler l'indentation du code dans le bloc . . . . .	77
15.3.4. Les numéros de légende (callouts) . . . . .	78
<b>16. Inclure des fichiers de code . . . . .</b>	<b>81</b>
16.1. Le bloc de code : rappel . . . . .	81
16.2. Inclure un fichier de code . . . . .	82
16.3. Inclure une partie d'un fichier de code . . . . .	84
16.3.1. Inclure une seule région de code . . . . .	84
16.3.2. Inclure plusieurs régions de code . . . . .	87
16.3.3. Exclure des régions enfant . . . . .	87
<b>17. Les tableaux . . . . .</b>	<b>90</b>
17.1. Comment travailler ce chapitre ? . . . . .	90
17.2. Déclarer un tableau et ses dimensions . . . . .	90
17.2.1. Déclarer le début et la fin d'un tableau . . . . .	90
17.2.2. Déterminer le nombre de colonnes . . . . .	90
17.2.3. Ajouter des lignes . . . . .	91
17.2.4. Agir sur la largeur des colonnes . . . . .	93
17.2.4.1. Comprendre la notion de largeur disponible . . . . .	93
17.2.4.2. Ajustement manuel de la largueur des colonnes . . . . .	93
17.2.4.3. Ajustement automatique en fonction du contenu . . . . .	94
17.2.5. Agir sur la largeur du tableau . . . . .	95
17.3. Ajouter un titre au tableau . . . . .	96
17.3.1. Où écrire le titre du tableau ? . . . . .	96
17.3.2. Personnaliser l'étiquette du titre d'un tableau . . . . .	96
17.4. Définir l'entête et le pied de tableau . . . . .	97
17.4.1. Définir les entêtes de colonne . . . . .	97
17.4.2. Définir le pied de tableau . . . . .	98
17.5. Aligner le contenu d'une colonne . . . . .	99
17.5.1. Alignement horizontal au niveau d'une colonne . . . . .	99
17.5.2. Alignement vertical au niveau d'une colonne . . . . .	101
17.5.3. Combiner alignement horizontal et vertical au niveau d'une colonne . . . . .	102

17.6. Aligner le contenu au niveau d'une cellule .....	103
17.7. Centrer le tableau dans la page .....	105
17.8. Mise en forme du contenu au niveau d'une colonne .....	106
17.9. Mise en forme au niveau du contenu d'une cellule. ....	111
17.10. Les bordures de tableau.....	113
17.10.1. Les bordures qui entourent le tableau .....	113
17.10.2. Les bordures entre les cellules .....	114
17.11. Ombrer une ligne sur deux .....	116
17.12. Créer un tableau à partir de données au format CSV .....	117
17.12.1. Générer un tableau à partir de données csv .....	117
17.12.2. Générer un tableau à partir d'un fichier csv.....	119
17.13. D'autres fonctionnalités sur les tableaux.....	120
<b>18. Modifier l'orientation du document ou d'une page.....</b>	<b>121</b>
<b>19. Les attributs de document .....</b>	<b>124</b>
19.1. Un attribut, c'est quoi ? .....	124
19.2. Manipuler un attribut .....	125
19.2.1. Déclarer / créer un attribut .....	125
19.2.2. Affecter une valeur à un attribut .....	126
19.2.3. Afficher la valeur d'un attribut.....	127
19.3. Les attributs intégrés.....	129
19.4. Les attributs de l'utilisateur.....	131
19.4.1. Qu'est-ce qu'un attribut de l'utilisateur ?.....	131
19.4.2. Des attributs de l'utilisateur pour ne pas se répéter .....	132
19.4.3. Des attributs de l'utilisateur pour conditionner le contenu à rendre .....	132
19.5. Désactiver un attribut .....	135
<b>20. Les compteurs .....</b>	<b>137</b>
<b>21. Les annexes d'un document .....</b>	<b>139</b>
21.1. Créer des parties pour les annexes .....	139
21.2. Personnaliser l'étiquette des annexes.....	142
<b>22. Le type de document .....</b>	<b>144</b>
22.1. Choisir le type de document de sortie.....	144
22.2. Spécifier le type de document.....	144
<b>23. Générer un index .....</b>	<b>146</b>
23.1. Qu'est-ce qu'un index ? .....	146
23.2. Afficher l'index .....	146
23.3. Ajouter des entrées à l'index .....	147
<b>24. Afficher / masquer du texte avec des conditions .....</b>	<b>150</b>
24.1. Conditionner l'affichage à l'existence d'un attribut .....	150
24.2. Conditionner l'affichage à la valeur d'un attribut .....	155
<b>25. Les bonnes pratiques de création d'un document AsciiDoc .....</b>	<b>160</b>
25.1. Un fichier AsciiDoc ne doit pas être "long" .....	160

25.1.1. Les inconvénients d'un fichier trop "long" .....	160
25.1.2. Quelle est la bonne "longueur" d'un fichier AsciiDoc ? .....	160
25.2. Créer un document principal à partir de plusieurs autres .....	161
25.3. Adopter une structure efficace par document .....	166
25.4. Partager un fichier pour ne pas se répéter .....	169
25.4.1. Partager un fichier qui ne contient pas d'images .....	169
25.4.2. Partager un fichier qui contient des images .....	171
25.5. Versionner son travail .....	174
<b>26. Le fichier de thème PDF .....</b>	<b>175</b>
26.1. Qu'est-ce qu'un fichier de thème PDF ? .....	175
26.2. Mettre en place son propre fichier de thème PDF .....	175
26.2.1. Placer le fichier de thème au même niveau que le fichier AsciiDoc .....	175
26.2.2. Un seul fichier de thème pour tous les fichiers asciidoc ! .....	178
26.2.3. Rendre variable le chemin vers le fichier de thème .....	179
26.2.4. Rendre variable le nom du fichier de thème .....	181
<b>27. Mise en forme à l'aide de rôles personnalisés .....</b>	<b>183</b>
27.1. Rappel sur les possibilités intégrées de formatage du texte .....	183
27.2. Créer un rôle personnalisé .....	183
<b>28. L'entête et le pied de page du pdf .....</b>	<b>187</b>
28.1. Rendu par défaut .....	187
28.2. Compléter les métadonnées (auteur, date de révision, ...) .....	188
28.3. Personnaliser l'entête de page du fichier pdf .....	189
28.4. Personnaliser le pied de page du fichier pdf .....	193
<b>29. La page de garde .....</b>	<b>195</b>
29.1. La page de garde par défaut .....	195
29.2. Personnaliser la page de garde .....	196
29.2.1. Ajout d'un logo à la page de garde .....	196
29.2.2. Ajout d'une image de fond sur la page de garde .....	199
29.2.3. Mise en forme des éléments textuels de la page de garde avec un fichier de thème ..	201
29.3. Besoin d'une page de garde complexe .....	204
<b>Index .....</b>	<b>209</b>

# 1. Préparation de l'IDE pour éditer du contenu AsciiDoc

Version 1.0.1 | Dernière mise à jour : 26/08/2024 à 11:17 | Auteur : Emmanuel Ravrat

Durée de réalisation : 30min

chemin : chapters|asciidoc|installation\_plugin\_asciidoc|main.adoc

## 1.1. Crédit d'un premier document AsciiDoc dans l'IDE



Si vous utilisez un autre IDE qu'un IDE Jetbrains, il vous appartient de trouver les fonctionnalités et paramétrages équivalents à ceux qui sont abordés dans ce support. Il se peut parfois qu'il n'y ait pas d'équivalent ou que les fonctionnalités ne soient pas aussi abouties.

Le langage AsciiDoc à l'instar de tous les langages à balisage léger peut être écrit dans n'importe quel éditeur de texte. Même le cultissime bloc-note de Windows peut faire l'affaire !

Toutefois, l'objectif est de gagner en productivité et quoi de mieux que d'utiliser son IDE favori pour écrire tous ses documents.

Lancez IntelliJ IDEA et créez un nouveau projet nommé "asciidoc" en suivant les recommandations ci-dessous.

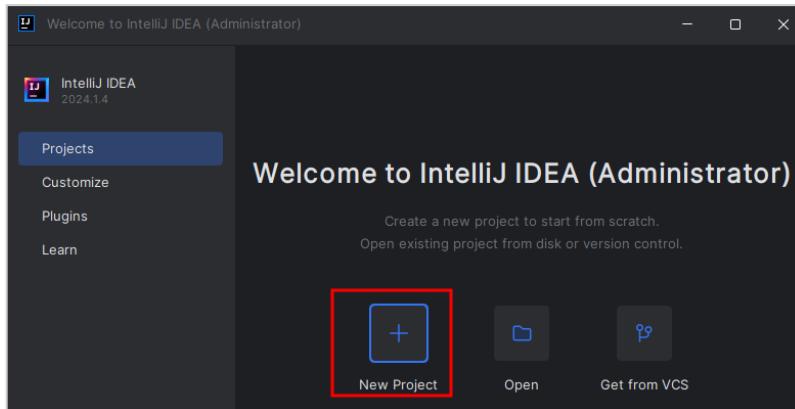


Figure 1. fenêtre de démarrage d'IntelliJ IDEA

Choisissez un projet vide, nommez le projet "asciidoc" et choisissez le répertoire de votre machine dans lequel il sera créé.

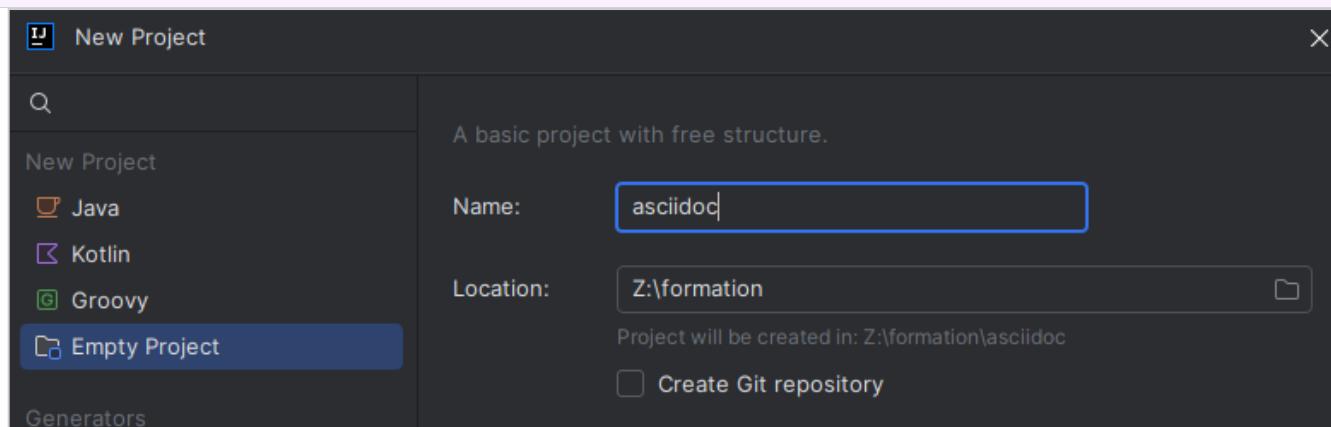


Figure 2. création d'un projet vide

A partir de maintenant, veillez à ne jamais utiliser de caractères spéciaux, de caractères accentués et des espaces dans le nom des fichiers et des dossiers que vous créerez sur votre machine.

Cela peut entraîner divers problèmes :

- certains systèmes de fichiers peuvent ne pas supporter ou mal gérer ces caractères
- les scripts ou outils en ligne de commande qui manipulent ces dossiers et fichiers rencontreront moins de problèmes
- certains protocoles de transfert de fichiers peuvent mal gérer certains caractères
- la portabilité entre différents systèmes est meilleure
- certains caractères spéciaux peuvent être utilisés de manière malveillante
- la lisibilité est facilitée



Vous devez réfléchir l'organisation de vos dossiers de travail en fonction des matières, des cours abordés, etc. C'est extrêmement important pour mieux s'y retrouver et faciliter les accès futurs à vos documents.

Un dossier "fourre tout" doit être banni de vos pratiques. Chacun de vos dossiers et fichiers doivent avoir un nom significatif qui permet d'identifier facilement son contenu. Nous reviendrons sur ce point dans un prochain cours.

Une fois le projet créé, vous pouvez avoir le message suivant :

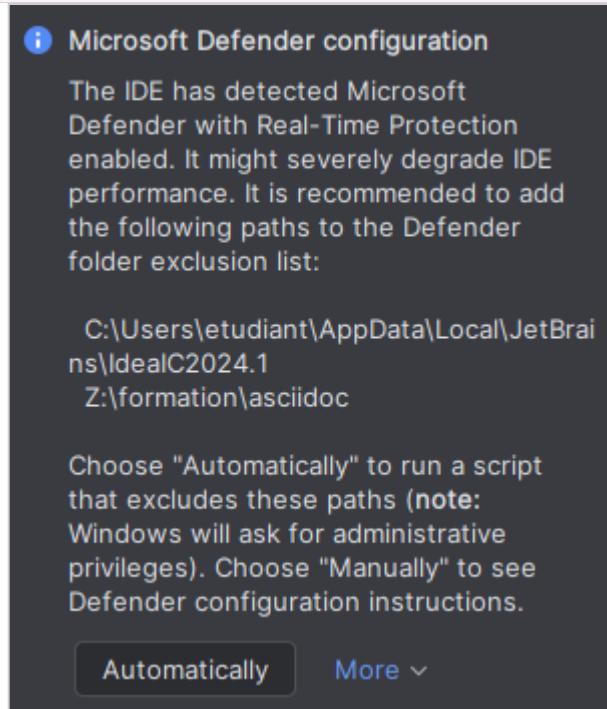


Figure 3. message à propos de Microsoft Defender

Je vous conseille de cliquer sur **[ Automatically ]** afin d'exclure la protection en temps réel pour IntelliJ sans quoi les performances de l'IDE peuvent être fortement dégradées.

Votre projet contient un dossier `.idea` et un fichier `asciidoc.iml`:

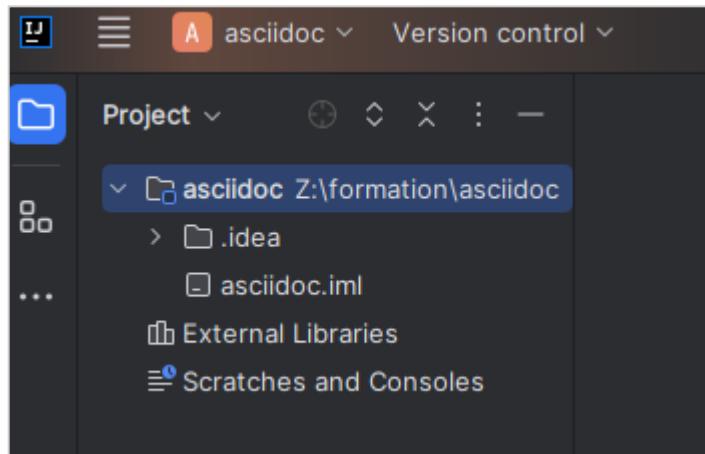


Figure 4. structure de base du projet créé

- le dossier `.idea` : ce dossier contient la configuration de l'IDE pour le projet ouvert (disposition des fenêtres, etc.)
- le fichier `asciidoc.iml` permet d'organiser vos modules. Cela est utile dans un projet Java mais dans dans le cas présent.

Faites un clic droit sur le dossier racine `asciidoc` du projet, puis **New > File**. Nommez le fichier `test_plugin_asciidoc.adoc`. L'extension `adoc` sera l'extension de tous nos fichiers asciidoc. C'est l'extension la plus répandue pour indiquer qu'il s'agit de texte au format asciidoc (on peut trouver les extensions `ad`, `asc`, `asciidoc`).

Commencez par écrire le contenu suivant dans le fichier qui s'est ouvert dans votre IDE :

= Ceci est le titre du document

Le caractère '=' permet de préciser que le texte qui le suit est le titre principal du document.

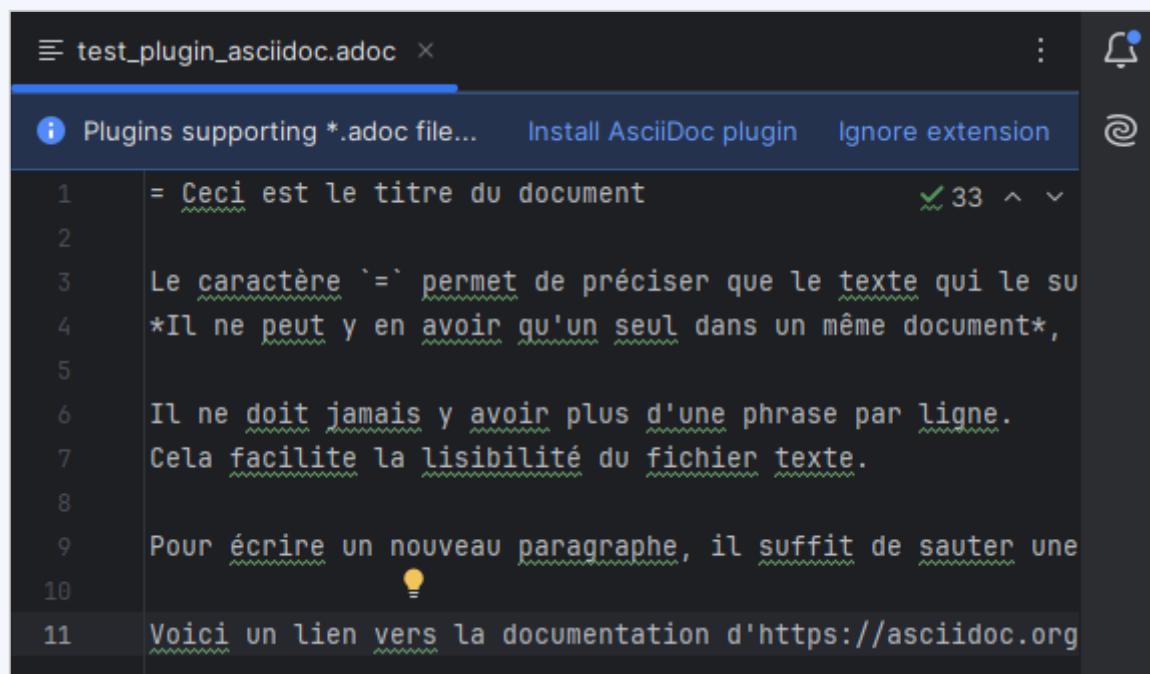
\*Il ne peut y en avoir qu'un seul dans un même document\*, que celui-ci soit constitué d'un seul ou de plusieurs fichiers asciidoc.

Il ne doit jamais y avoir plus d'une phrase par ligne.  
Cela facilite la lisibilité du fichier texte.

Pour écrire un nouveau paragraphe, il suffit de sauter une ligne et de commencer une nouvelle phrase.

Voici un lien vers la documentation d'[https://asciidoc.org/\[Asciidoc\]](https://asciidoc.org/[Asciidoc]).

En principe, le texte est écrit au kilomètre ce qui peut vous amener à ne pas voir toute la ligne :



```
= Ceci est le titre du document
Le caractère '=' permet de préciser que le texte qui le su
*Il ne peut y en avoir qu'un seul dans un même document*,

Il ne doit jamais y avoir plus d'une phrase par ligne.
Cela facilite la lisibilité du fichier texte.

Pour écrire un nouveau paragraphe, il suffit de sauter une
line.

Voici un lien vers la documentation d'<a href="https://asciidoc.org/[Asciidoc]>https://asciidoc.org/[Asciidoc]</a>.
```

Figure 5. illustration des lignes qui "dépassent" de la fenêtre

Vous pouvez modifier ce comportement en allant dans les paramètres de l'IDE :

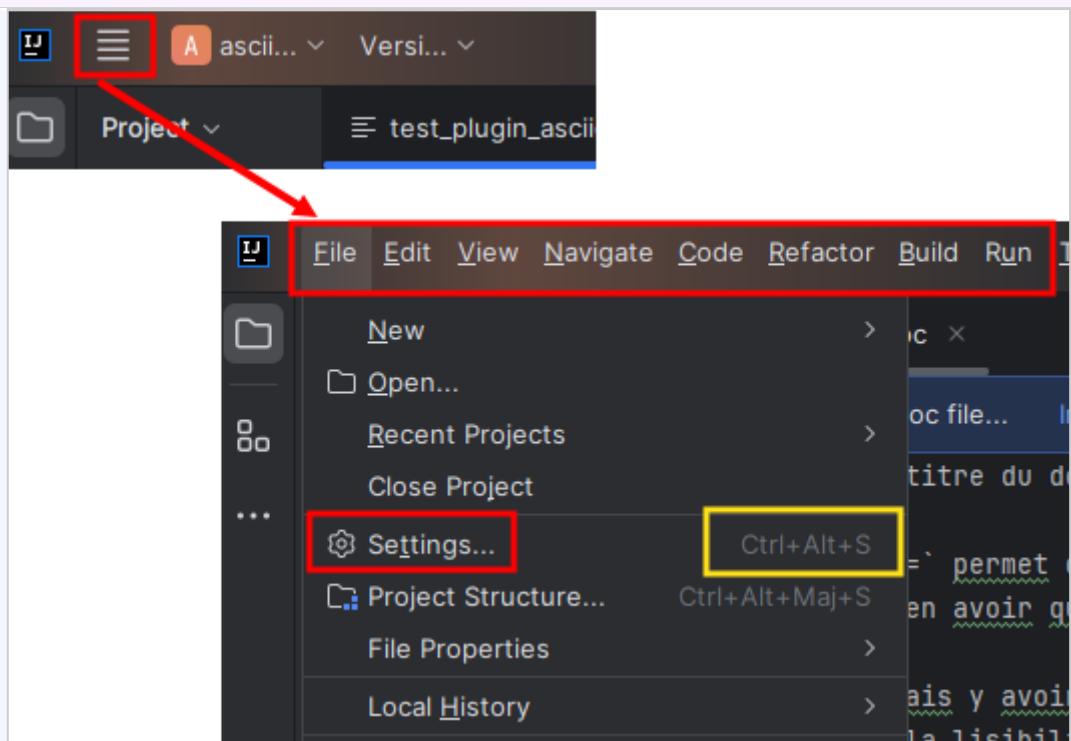


Figure 6. accès aux paramètres de configuration de l'IDE

Vous pouvez remarquer qu'un raccourci clavier est déjà affecté **CTRL + ALT + S** pour ouvrir la fenêtre des paramètres.

Dans le champ de recherche, saisissez **soft wrap** puis **Editor > General|Soft-wrap these files**

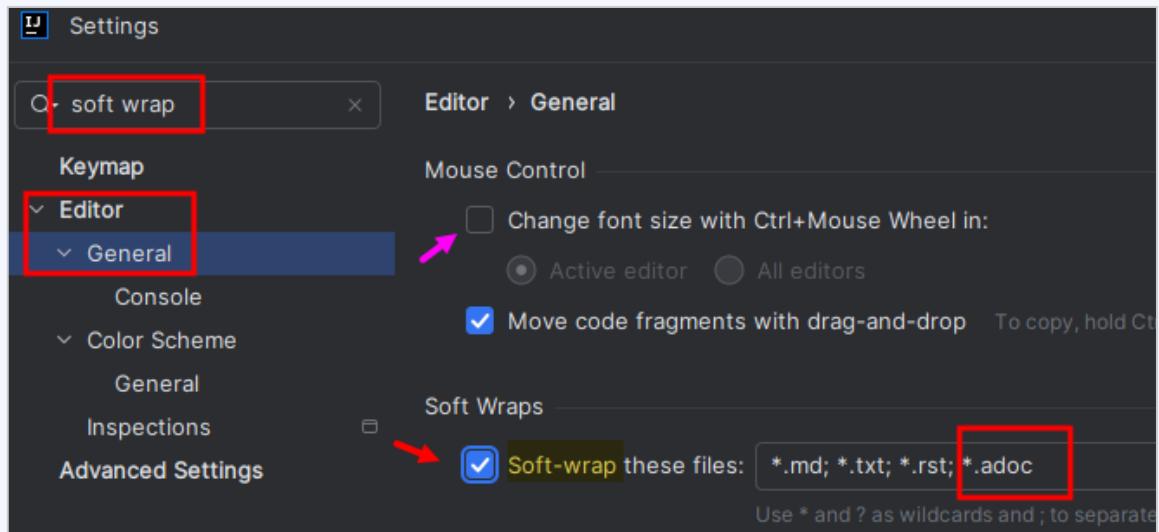


Figure 7. paramétrage du "retour sous la ligne" en cas de dépassement

Profitez-en pour activer la possibilité de modifier la taille de la police via un **CTRL + Molette souris** :

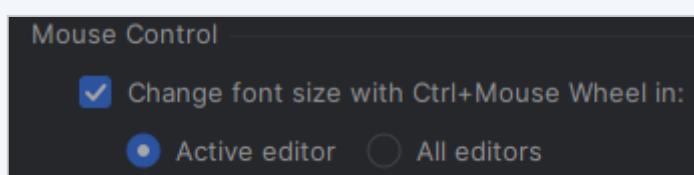


Figure 8. activation de la possibilité de zoomer la taille de police avec la souris

Maintenant, les lignes sont affichées de façon à ne pas dépasser de la fenêtre. Lorsqu'une ligne est "tronquée", une petite flèche indique que la ligne continue en-dessous. Cependant, il s'agit bien d'une seule et même ligne si vous regardez son numéro.

```
= Ceci est le titre du document
Le caractère '=' permet de préciser que le texte qui
le suit est le titre principal du document.
*Il ne peut y en avoir qu'un seul dans un
document*, que celui-ci soit constitué
de plusieurs fichiers AsciiDoc.
Il ne doit jamais y avoir plus d'une phrase par ligne.
Cela facilite la lecture du fichier texte.
Pour écrire un nouveau paragraphe, il suffit de sauter
une ligne et de commencer une nouvelle phrase.
Voici un lien vers la documentation d'<a href="https://asciidoc.org/[Asciidoc]>https://asciidoc.org/[Asciidoc]</a>.
```

Figure 9. symbole représentant un retour sous la ligne

## 1.2. Installation du plugin Asciidoc

Pour l'instant, il n'y a pas de différence entre le fait d'éditer un fichier asciidoc dans notre IDE ou dans le bloc-note de Windows. Là où l'utilisation de l'IDE est intéressante, c'est qu'il est possible d'avoir une prévisualisation du rendu au fur et à mesure de l'écriture dans le fichier.

Vous avez dû remarquer un message indiquant qu'un plugin supportant les fichiers `adoc` a été trouvé.

Figure 10. message affiché dans l'IDE indiquant qu'un plugin existe pour le fichier adoc

Nous pourrions cliquer directement sur "Install AsciiDoc plugin" pour gagner du temps. Cependant, nous allons voir comment faire cela via les paramètres de configuration.

Ouvrez la fenêtre des paramètres de configuration `CTRL + ALT + S` (ou **File > Settings**, saisissez "plugin" dans le champ de recherche. Vous avez deux onglets : "Marketplace" et "Installed". Le

premier permet de rechercher des plugins et le second liste tous les plugins déjà installés.

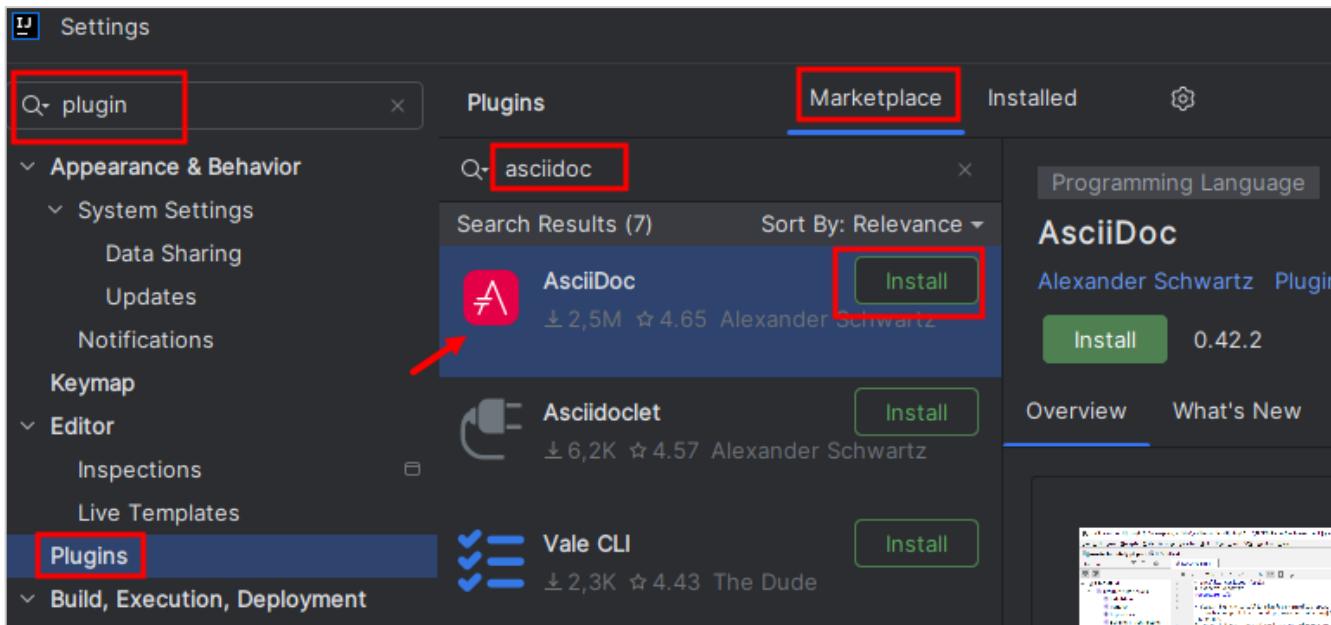


Figure 11. recherche et installation du plugin AsciiDoc

Cliquez sur [ Install ].

Un message va vous indiquer que le plugin ne provient pas de JetBrains. Ainsi, il vous appartient de vérifier que le plugin est sûr. C'est le cas ici.

Si vous doutez, vous pouvez aller voir la page du plugin ou faire des recherches complémentaires.

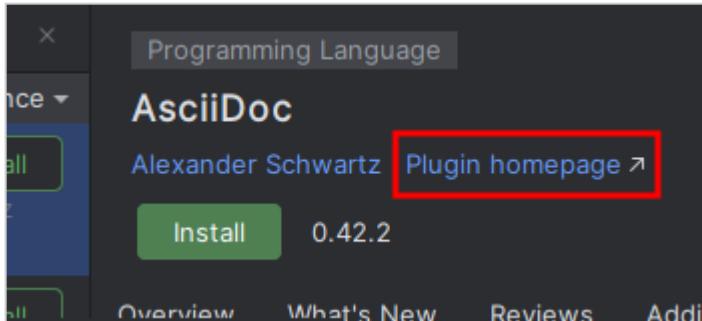


Figure 12. chaque plugin est normalement lié à une page dédiée

Si vous ne l'avez pas déjà fait, cliquez sur [ Install ] puis sur [ Ok ]. Il peut vous être demandé de relancer l'IDE (cela n'a pas été le cas pour moi). Vous remarquerez que votre fenêtre d'édition afficher une partie à droite de votre fichier texte. Celle-ci permet de prévisualiser ce que vous écrivez :

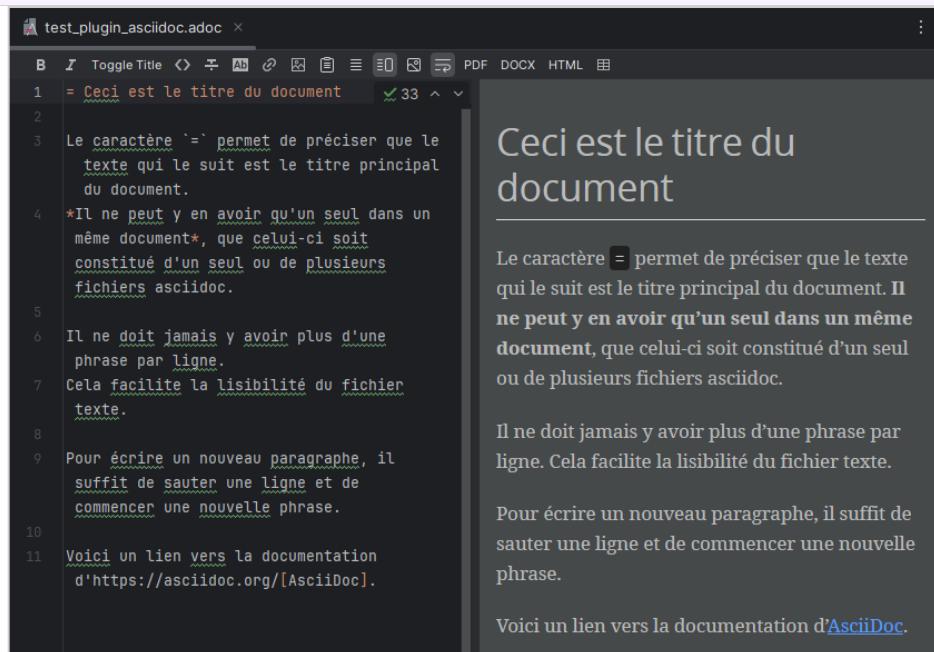


Figure 13. prévisualisation du contenu textuel grâce au plugin

Vous avez une barre d'outils basique qui s'affiche automatiquement dès lors que vous travaillez avec un fichier asciidoc.



Figure 14. barre d'outils d'édition de texte Asciidoc

Utilisez la barre d'édition afin de tester toutes ses possibilités sur du texte et profitez-en pour observer les caractères qui servent de balisage et leur rendu visuel.

## 1.3. Générer un fichier pdf à partir du fichier Asciidoc

La barre d'outil permet de convertir un fichier AsciiDoc en trois formats (pdf, HTML et docx).

Ce qui nous intéresse ici est la sortie au format pdf.

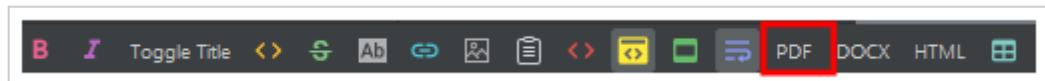


Figure 15. barre d'outils du plugin Asciidoc avec création d'un pdf

Effectivement, l'objectif est d'utiliser Asciidoc pour écrire des supports qui seront générés au format pdf.

Le format pdf peut être ouvert depuis n'importe quelle plateforme (Windows, Linux, MacOs, Androïd, etc.) et ne requiert aucune installation particulière. Un simple navigateur suffit à lire un fichier pdf. Ce n'est pas le cas du format docx qui nécessite Word ou l'utilisation d'un logiciel de traitement de texte.

Le format HTML aurait pu être utilisé comme format de sortie pour les mêmes raisons. Cependant, le format pdf permet de se repérer plus facilement avec une pagination et est plus facilement distribuable.

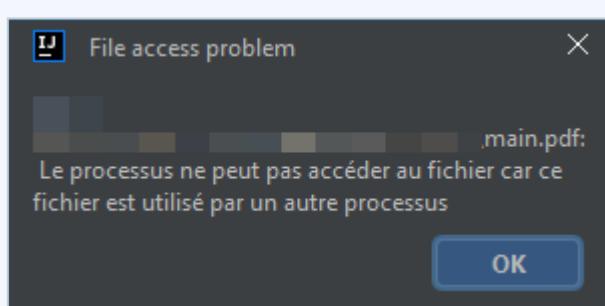
Le fichier pdf généré sera construit à partir du code AsciiDoc depuis lequel le bouton [ PDF ] aura été cliqué.



Dans l'avenir, vous allez très probablement inclure des fichiers (voir ce point de cours : [\[title\\_inclure\\_des\\_fichiers\\_asciidoc\]](#) dans le chapitre [chapitre 25, Les bonnes pratiques de création d'un document AsciiDoc](#)). Il arrivera que vous cliquez sur le bouton [ PDF ] depuis un fichier inclus. Cela ne générera que le fichier pdf du fichier inclus. Vous saurez alors que vous devez générer le document depuis le fichier asciidoc principal.

Lorsque vous générerez un fichier pdf depuis un fichier AsciiDoc, il est probable que celui-ci soit automatiquement ouvert dans votre liseuse pdf.

Il faut penser à fermer le fichier pdf avant de lancer une nouvelle génération d'un pdf à partir du même fichier. Sans cela, le fichier ne sera pas généré car le fichier pdf à écrire est déjà ouvert :



J'utilise très souvent le terme de "rendu" lorsque j'aborde le résultat visuel de la génération au format pdf.

## 2. Gestion de la langue française dans l'IDE

Version 1.0.1 | Dernière mise à jour : 26/08/2024 à 11:38 | Auteur : Emmanuel Ravrat

Durée de réalisation : 25min

### 2.1. Installation d'un outil de vérification du texte



Il est préférable d'avoir installé le plugin AsciiDoc avant de continuer.

A l'instar des logiciels Word, Writer, etc., il est possible de profiter d'une analyse du contenu écrit en temps réel. C'est-à-dire que les fautes d'orthographe, de syntaxe, de conjugaison peuvent être détectées et mise en valeur dans l'IDE.

Les plugins Grazie-Lite et Grazie Pro permettent d'ajouter une vérification du texte écrit. Le premier devrait déjà être installé par défaut. Le second peut être installé dans sa version gratuite.

Grazie contient un dictionnaire qui permet de vérifier l'orthographe des mots et cela pour plusieurs langues !

A ce stade, de nombreux mots sont encore soulignés malgré l'installation du plugin. Effectivement, pour l'instant, c'est la langue anglaise qui est vérifiée. Les mots français sont donc reconnus comme étant mal orthographiés.

En principe, les mots qui étaient soulignés ne devraient plus l'être (sauf ceux qui sont inconnus)



Attention, l'outil ne permet pas une vérification complexe de ce que vous écrivez. Il vous appartient d'appliquer les règles d'orthographe, de grammaire et de conjugaison que vous avez apprises depuis votre plus tendre enfance. Alors, ne faites jamais l'impasse sur une relecture (et même plusieurs !).

J'entends déjà d'ici les "je suis nul et j'ai toujours été nul en orthographe, etc." Et bien sachez que tout le monde peut progresser en cherchant les règles et en s'efforçant à les appliquer.

J'ai personnellement rencontré beaucoup de difficultés sur ce point et à force de vérifier encore et encore l'application des règles, j'ai beaucoup progressé.

Il ne vous est pas demandé d'être infaillible mais de progresser dans votre expression !

## 2.2. Configuration de la langue à inspecter

A ce stade, de nombreux mots devraient encore être soulignés. il faut indiquer les langues à vérifier. Par défaut, seul l'anglais est pris en charge.

Dans les paramètres de configuration **CTRL + ALT + S**, saisissez le mot **natural** et cliquez sur "Natural Languages". Puis cliquez sur l'icone **+** et choisissez le langage "Français" :

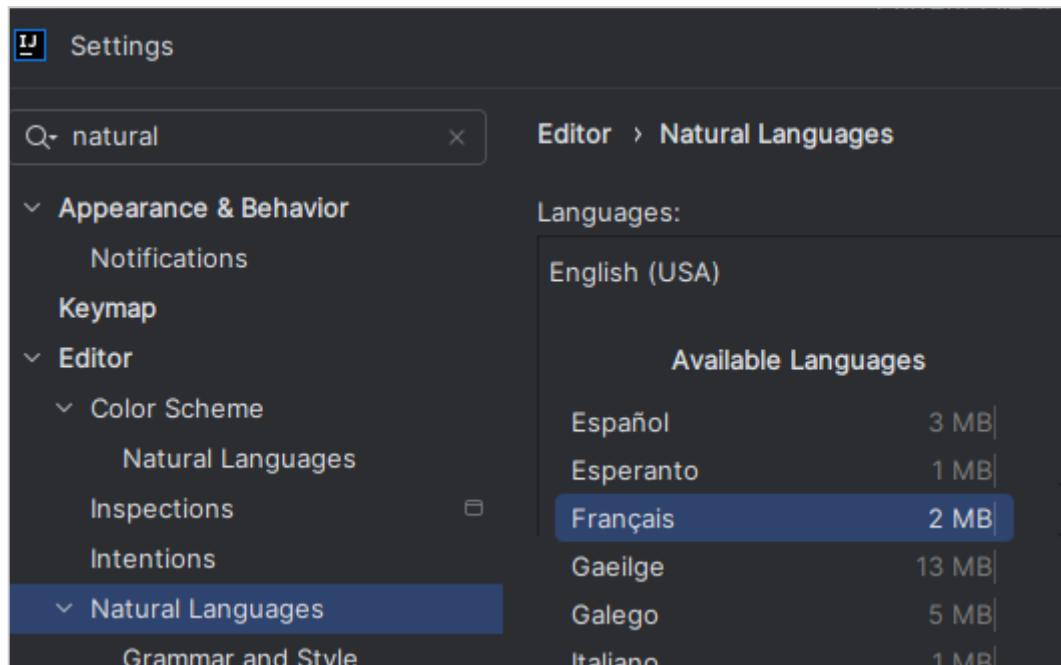


Figure 16. ajout de la langue française à l'éditeur

En principe, les mots ne sont plus soulignés dans l'éditeur !

## 2.3. Ajout de dictionnaires

Si vous avez des mots qui restent soulignés alors qu'ils ne le devraient pas, vous pouvez éventuellement ajouter des dictionnaires supplémentaires. Cela peut être le cas pour des noms propres, des termes techniques, etc.

La seule difficulté est de trouver un dictionnaire. Un dictionnaire est un fichier texte contenant une simple liste de mots. Il peut être ouvert avec n'importe quel éditeur de code, y compris le bloc-note.

Pour l'exemple, vous pouvez télécharger le [dictionnaire Gutenberg de 1998](#).

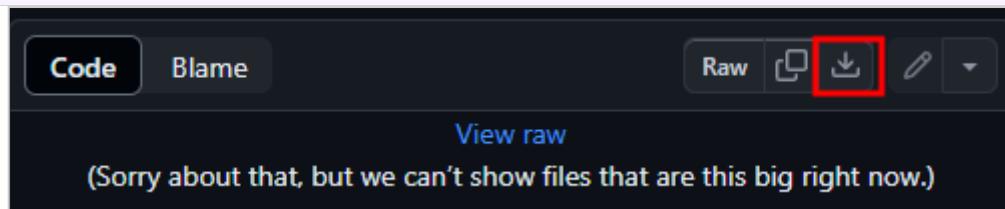


Figure 17. récupération du dictionnaire Gutenberg

Il s'agit d'un simple fichier texte contenant une liste de 336531 mots.

Dans les paramètres de configuration **CTRL + ALT + S**, saisissez dans le champ de recherche le mot "spelling".

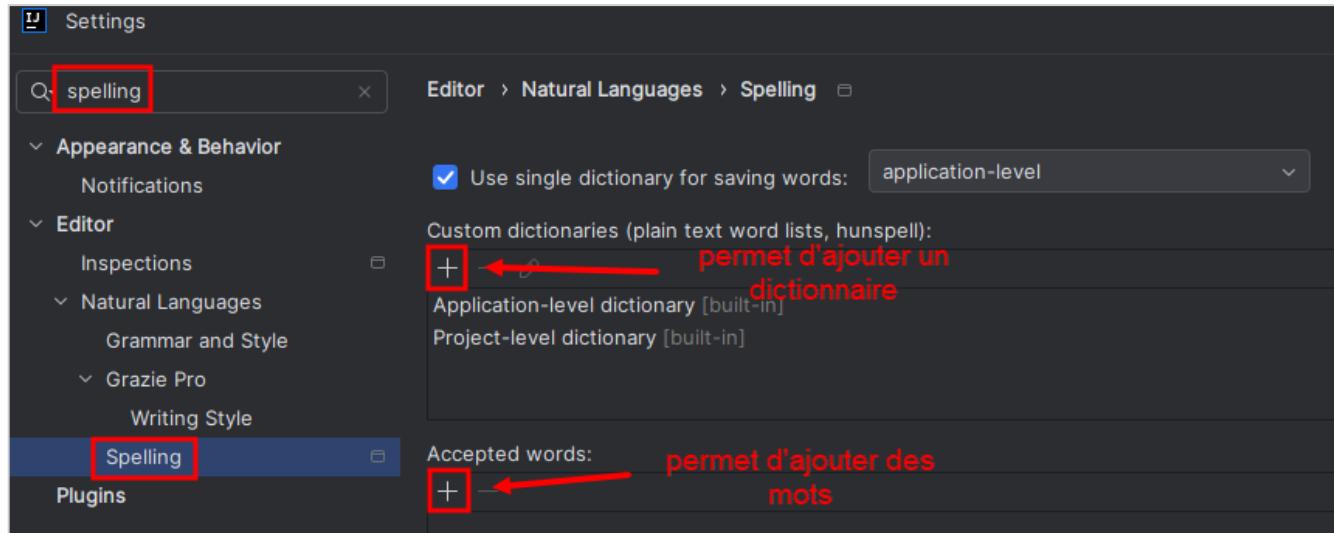


Figure 18. ajouter des dictionnaires et ou des mots

Pour ajouter le dictionnaire, il suffit de préciser l'emplacement de votre fichier de dictionnaire. Je recommande de créer un dossier dans lequel vous placez tous vos dictionnaires au même endroit.

Il est possible d'ajouter manuellement un mot depuis la fenêtre d'édition.

Par exemple, j'ai écrit le mot **tradisateur** qui est souligné car inconnu. Au **survol de ce mot**, une petite fenêtre vous fait une proposition de remplacement ou vous propose "More actions...". Cela vous permet de l'ajouter manuellement :

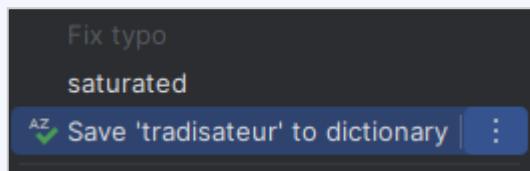


Figure 19. ajout d'un mot au dictionnaire de l'application

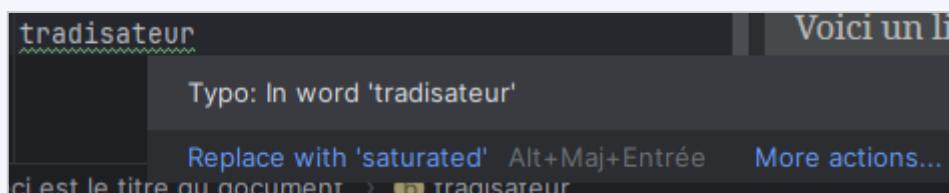


Figure 20. ajout d'une entrée dans le dictionnaire de l'IDE

Le mot n'est plus souligné et figure bien dans la liste des mots ajoutés dans les paramètres de configuration.

# 3. Utiliser son navigateur pour prévisualiser un fichier AsciiDoc

Version 1 | Dernière mise à jour : 23/07/2024 à 19:46 | Auteur : Emmanuel Ravrat

Durée de réalisation : 25min

## 3.1. Installation du plugin Asciidoctor dans son navigateur

S'il est possible d'utiliser un environnement de développement intégré avec un plugin pour pouvoir visualiser un document écrit en AsciiDoc, il est également possible de voir le rendu dans son navigateur, rendu qui est vraiment très propre !

Les navigateurs Chrome, Firefox, Edge et Opera disposent d'un plugin qui permet d'ouvrir un fichier AsciiDoc dans le navigateur.

1. Commencez par installer le plugin correspondant au navigateur que vous utilisez en cliquant sur le lien adéquat :

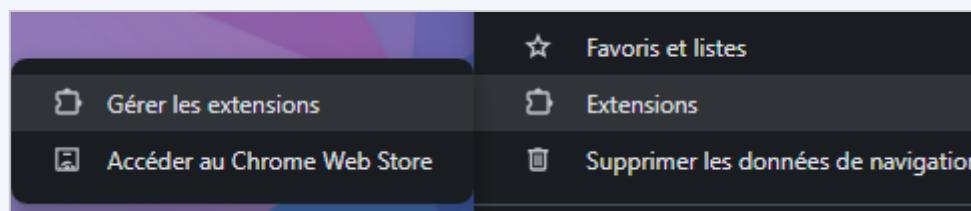
- [plugin pour Chrome](#)
- [plugin pour Firefox](#)
- [plugin pour Opera](#)
- [plugin pour Edge](#)
- pour Safari, il faut chercher dans le store (et m'indiquer s'il existe). A défaut, un dépôt github met à disposition une extension : <https://github.com/ggrosssetie/asciidoctor-safari-extension>. Je n'ai aucune idée des fonctionnalités prises en charge par cette extension.

2. Accordez l'accès au système de fichiers à l'extension :

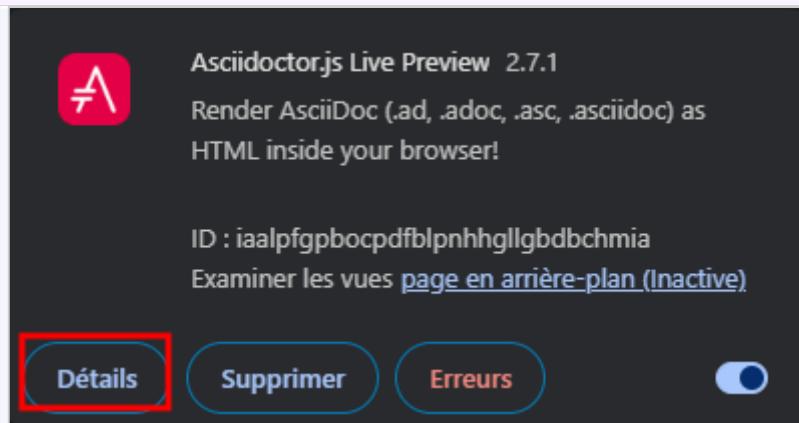
Pour Chrome, cliquez sur l'icône des paramètres



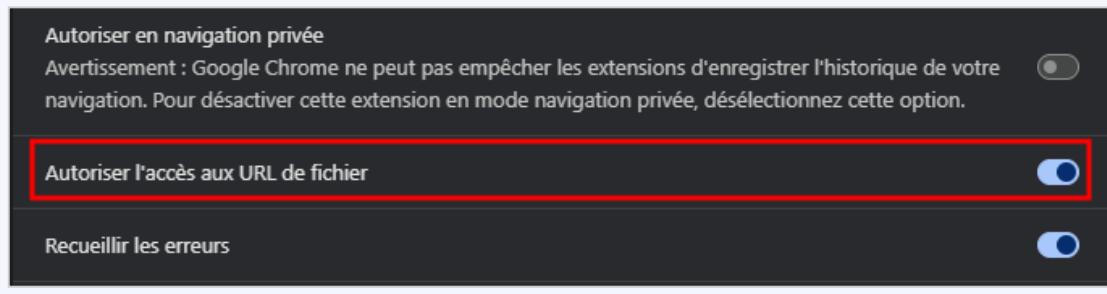
Cliquez sur Extensions puis sur Gérer les extensions :



Puis sur Détails au niveau de l'extension Asciidoctor.js :



Et enfin, il faut activer l'accès aux URL de fichier



Si le plugin ne peut pas avoir accès au système de fichiers, le navigateur affichera le code source de votre fichier AsciiDoc.

3. Pour visualiser un fichier AsciiDoc dans le navigateur, il suffit de le glisser sur un onglet déjà ouvert ou de faire un clic droit et d'ouvrir le fichier avec votre navigateur.

## 3.2. Les options du plugin de rendu Asciidoc

Dans la gestion des extensions de votre navigateur **Extensions > Gérer les extensions**, vous pouvez configurer des options pour l'extension "Asciidoctor.js".

Cliquez sur l'icône de lien externe au niveau des options de l'extension :

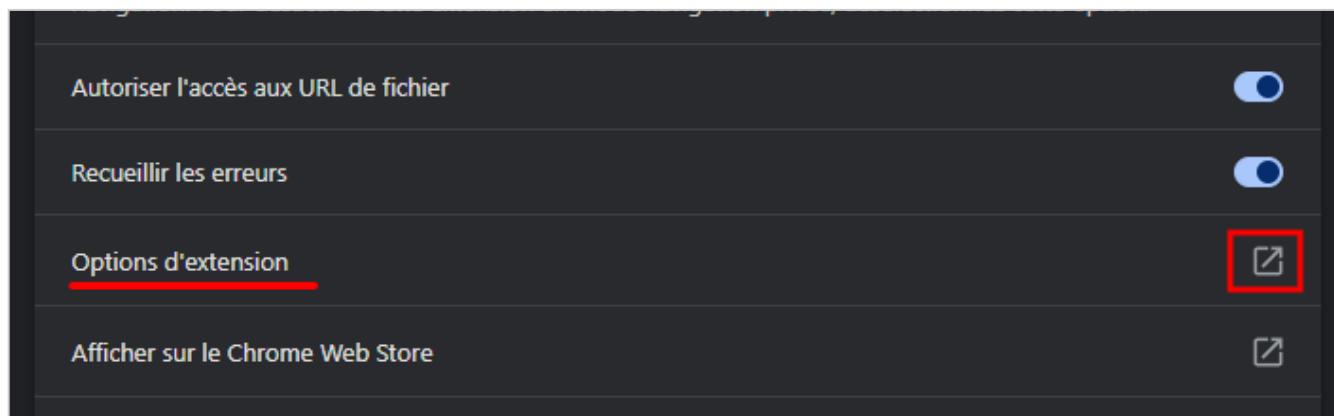


Figure 21. accès aux options de l'extension dans Chrome

Celles-ci vous permettent de personnaliser le rendu du document AsciiDoc entre autres nombreuses choses.



Figure 22. option de personnalisation des attributs et de la sécurité

**Poll for changes**

To files loaded from URLs

1s 

durée entre deux rafraîchissements (utile pour éditer le

To local files

1s 

Local files use the file protocol (or no protocol)

utilise si le fichier Asciidoc contient des

**Diagrams extension**

diagrammes (sera abordé ultérieurement)

Enable the diagrams extension

The diagrams extension is sending the text diagrams to an instance of Kroki to display them as images in the preview. Kroki is a free open-source project.

**Server URL**

https://kroki.io 

By default the diagram extensions sends your diagrams to the free public cloud instance https://kroki.io but you can install Kroki on your own infrastructure. Once you've installed Kroki, make sure to update the server URL to point to your instance.

Figure 23. options de rechargement de la page et de prise en charge des diagrammes

**Additional resources**

**Theme / Stylesheet**

colony 

choix du thème pour le rendu dans le navigateur

The CSS styles to apply to the rendered HTML.

 Add a stylesheet...

**JavaScript**

</> 

Add a custom JavaScript to the rendered HTML.

 Add a JavaScript...

Load the JavaScript...

... after the document has been rendered

... before the document has been rendered

Figure 24. options de personnalisation du rendu

# 4. Les commentaires

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 12:03 | Auteur : Emmanuel Ravrat

Durée de réalisation : 15min

Les **commentaires AsciiDoc** sont des lignes de texte qui sont ignorées dans le rendu final du document. C'est-à-dire qu'ils ne sont pas affichés dans le rendu final que ce soit un rendu html, pdf, etc.

Ils sont très utiles pour écrire des informations à destination de l'auteur, d'un relecteur, etc.

Code AsciiDoc	Rendu
<p><b>Ecrire un commentaire sur une ligne</b></p> <p>Un commentaire AsciiDoc commence par un double slash <code>//</code> :</p> <div style="border: 1px solid #ccc; padding: 10px;"><p>Chuck Norris ne peut pas mourir.</p><p>// il faut reformuler cette phrase car elle est redondante ①</p><p>Chuck Norris est immortel.</p></div> <p>① commentaire qui ne fait qu'une seule ligne. Il ne sera pas rendu.</p>	<p>Chuck Norris ne peut pas mourir.</p> <p>Chuck Norris est immortel.</p>
<p><b>Ecrire un commentaire multiligne</b></p> <p>Pour écrire un commentaire multiligne, il faut utiliser 4 slashes <code>///</code> au début du bloc de commentaire puis à la fin.</p> <div style="border: 1px solid #ccc; padding: 10px;"><p>1 Chuck Norris est le goat !</p><p>2 ////①</p><p>3</p><p>4 Je peux écrire ce que je veux,</p><p>5 il reste invisible (sauf pour Chuck !)</p><p>6</p><p>7 ////③</p><p>8 Mais JCVD cherche à le défier !</p><p>9 RIP petit JCVD.</p></div> <p>① début du commentaire</p> <p>② fin du commentaire</p>	<p>Chuck Norris est le goat !</p> <p>Mais JCVD cherche à le défier ! RIP petit JCVD.</p>

Code AsciiDoc	Rendu
<p><b>Ecrire un commentaire multiligne dans un bloc</b></p> <p>Cette technique utilise un bloc ouvert. C'est un type de bloc délimité par des doubles tirets <code>--</code> et qui est marqué par l'attribut de bloc <code>comment</code> :</p> <pre data-bbox="165 467 959 817">1 Chuck Norris est le goat ! 2 [comment] ① 3 -- 4 Un bloc de commentaire 5 6 Je peux écrire ce que je veux, il reste invisible ! 7 -- 8 Mais JCVD cherche à le défier ! 9 RIP petit JCVD.</pre> <p>① l'attribut de bloc <code>comment</code> indique que le bloc doit être traité comme un bloc de commentaire. Il doit donc être ignoré dans le rendu.</p> <p>② début du bloc ouvert</p> <p>③ fin du bloc ouvert</p>	<p>Chuck Norris est le goat !</p> <p>Mais JCVD cherche à le défier ! RIP petit JCVD.</p>



Les commentaires sont un allié précieux dans la rédaction de document. Il ne faut pas hésiter à en abuser !

Si vous rencontrez un besoin particulier ou une problématique particulière concernant les commentaires, vous pouvez lire la [page de la documentation dédiée aux commentaires](#).

# 5. Les titres

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 13:55 | Auteur : Emmanuel Ravrat

Durée de réalisation : 30min

## 5.1. L'importance des titres

Un document bien conçu se doit d'être organisé en parties. Chaque partie doit être annoncée par un **titre**, **sous-titre**, etc.

Ces titres permettent de structurer le document et facilitent la navigation dans celui-ci. C'est pourquoi chaque titre doit être simple, explicite et concis. Il est important de réfléchir à leur formulation, notamment parce que les titres vont donner lieu à une table des matières.

Une **table des matières** est une liste qui montre les titres des différentes parties d'un document, avec les numéros de page associés. Elle aide le lecteur à trouver rapidement le contenu qu'il cherche.

La lecture de la table des matières doit permettre de donner une idée assez précise du contenu.

## 5.2. Le titre principal d'un document AsciiDoc

Chaque document AsciiDoc doit avoir un titre principal (comme le titre d'un livre, d'un rapport, d'un compte rendu, d'un résumé, etc.)

Il ne peut y avoir qu'un seul titre principal dans un document AsciiDoc.

Un **titre** est indiqué avec le caractère `=` suivi d'un espace. Le nombre de caractères `=` détermine le niveau du titre.

Imaginez que vous voulez écrire un document sur Chuck Norris



Vous voulez donner le titre « Chuck Norris: l'homme, le mythe, la légende »"

= Chuck Norris: l'homme, le mythe, la légende ①

① Le titre principal est indiqué par la présence d'un seul caractère `=` suivi d'un espace

**Le titre principal ne doit jamais être précédé d'une ligne vide ou de texte à afficher.**



Une ligne vide est du "texte" dans le sens où elle doit être affichée. Or, aucun texte à afficher ne doit être écrit avant le titre. Cela signifie qu'il est possible d'écrire des commentaires et de déclarer des attributs avant le titre principal puisque ces éléments ne constituent pas du texte à afficher.



**Le contenu textuel qui vient après le titre principal doit toujours être précédé d'une ligne vide.**

## 5.3. La structure du document

Il est très important de structurer ses documents, d'autant plus s'ils sont longs et / ou couvrent plusieurs sujets.

Un document structuré avec des titres va vous permettre de générer automatiquement une table des matières avec pagination.



Le lecteur d'un document doit être en mesure de comprendre son contenu rien qu'en lisant la table des matières, donc la structure du document.

Ce qui structure un document est la **hiérarchie** des titres.

Complétons le titre principal en ajoutant les chapitres (ou parties) qui composent le document :

```
1 = Chuck Norris: l'homme, le mythe, la légende
2
3 == L'origine de la légende
4
5 == Pouvoirs surnaturels
6
7 == Aventures épiques
8
9 == Influence culturelle globale
```

Le document à un titre principal identifié par le caractère `=`.

Il est composé de 4 parties / chapitres identifiables par le double égal `==`. Le double égal permet d'indiquer qu'il s'agit d'une sous partie au titre principal.



**Un titre, sous-titre, sous sous-titre, etc. doit toujours être suivi d'une ligne vide.**

Nous pouvons affiner le contenu de notre document en ajoutant les sous-parties des chapitres. Puisque les chapitres sont marqué par un double égal ==, leurs sous-parties seront marquées avec trois signes égal ===:

```
1 = Chuck Norris: l'homme, le mythe, la légende
2
3 == L'origine de la légende
4
5 === Enfance extraordinaire
6
7 === Les débuts de la force
8
9 == Pouvoirs surnaturels
10
11 === Force absolue
12
13 === Maîtrise informatique
14
15 == Aventures épiques
16
17 === Rencontres avec la faune
18
19 === Exploits philosophiques
20
21 == Influence culturelle globale
```

Grâce aux caractères =, on distingue bien l'imbrication des parties entre elles. C'est ce que l'on appelle la hiérarchie des titres.

Allons encore plus loin dans la structure de notre document en ajoutant quand cela est nécessaire des sous-parties dans les parties identifiées par trois signes égal ===. Ces sous-parties sont marquées par 4 signes égal ===== :

```
1 = Chuck Norris: l'homme, le mythe, la légende
2
3 == L'origine de la légende
4
5 === Enfance extraordinaire
6
7 ===== Chuck Norris et les jouets
8
9 ===== Les premières victoires
10
11 === Les débuts de la force
12
13 ===== Chuck Norris vs. la gravité
14
15 ===== La naissance d'une légende
16
```

```
17 == Pouvoirs surnaturels  
18  
19 === Force absolue  
20  
21 ===== Chuck Norris et les montagnes  
22  
23 ===== Sa main, plus forte que l'acier  
24  
25 === Maîtrise informatique  
26  
27 ===== Chuck Norris, le maître du code  
28  
29 ===== Quand l'ordinateur obéit à Chuck  
30  
31 == Aventures épiques  
32  
33 === Rencontres avec la faune  
34  
35 ===== Chuck Norris et les animaux sauvages  
36  
37 ===== Comment Chuck a dompté le dragon  
38  
39 === Exploits philosophiques  
40  
41 ===== Les pensées profondes de Chuck Norris  
42  
43 ===== Chuck Norris et la sagesse universelle  
44  
45 == Influence culturelle globale
```



Avec AsciiDoc, il faut toujours commencer à écrire une ligne contre la marge sans jamais créer de décalage. Cela est valable pour tout : les titres, les listes, les paragraphes, etc.

Je vais souvent faire référence à des niveaux de titre pour exprimer un titre avec un, deux, trois, quatre ou cinq caractères égal. Or, le titre de niveau 0 correspond au titre principal, le titre de niveau 1 correspond à un sous-titre au titre principal et ainsi de suite :



```
= titre de niveau 0  
== titre de niveau 1  
=== titre de niveau 2  
===== titre de niveau 3  
===== titre de niveau 4
```

```
===== titre de niveau 5
```

Vous aurez remarqué que le niveau de titre correspond au nombre de caractères = - 1.  
S'il y a 4 caractères =, le niveau de titre est 3.

Il y a 6 niveaux de titre maximum (de 0 à 5). Il ne peut y avoir qu'un seul titre de niveau 0.

## 5.4. La table des matières

### 5.4.1. Afficher la table des matières

Pour afficher la table des matières, il faut le préciser sous le titre principal du document à l'aide d'un **attribut**.

Un **attribut** est une variable qui permet de contrôler et de personnaliser le rendu d'un document (voir le [chapitre sur les attributs de document](#)). Un attribut peut être utilisé pour stocker une valeur réutilisable à différents endroits du document. Un attribut est un nom encadré de double points :**nom\_attribut**:. Un attribut peut ne pas avoir de valeur, mais lorsqu'il en a une, celle-ci est précisée après un espace : :**nom\_attribut**: chuck Norris.

La table des matières peut se traduire en anglais par « table of content » abrégée « toc ».

Voici comment afficher la table des matières de notre document sur Chuck Norris (à ne pas prononcer trop fort car il n'aime pas que l'on parle de lui...) :

```
1 = Chuck Norris: l'homme, le mythe, la légende
2 :toc: ①
3
4 //suite du document
5 == L'origine de la légende
6
7 //contenu....
```

① la simple présence de l'attribut :**toc**: sous le titre suffit à afficher la table des matières.

# Chuck Norris: l'homme, le mythe, la légende

## Table of Contents

L'origine de la légende

Enfance extraordinaire

Les débuts de la force

Pouvoirs surnaturels

Force absolue

Maîtrise informatique

Aventures épiques

Rencontres avec la faune

Exploits philosophiques

Influence culturelle globale

Figure 25. Table des matières générée avec les options par défaut

Vous remarquerez que tous les titres ne sont pas rendus. Par défaut, la table des matières affiche les titres jusqu'au niveau 2. Ce comportement est personnalisable et est abordé dans le point suivant.



L'attribut `toc` est un **attribut de document**. Les attributs de document permettent **d'agir sur le rendu du document** (ex : ajout d'éléments dans le document tels que la table des matières, un index, etc.)



Il peut y avoir plusieurs attributs de document sous le titre mais ils ne doivent jamais être séparés du titre par une ligne vide sans quoi ils sont inactifs.

Avant de continuer, il faut comprendre une notion importante : l'**entête de document asciidoc**.

L'**entête de document** est la partie du fichier AsciiDoc qui comprend le titre et les attributs de document qui sont déclarés sous celui-ci. J'utilise souvent ce terme, il faut bien le comprendre.



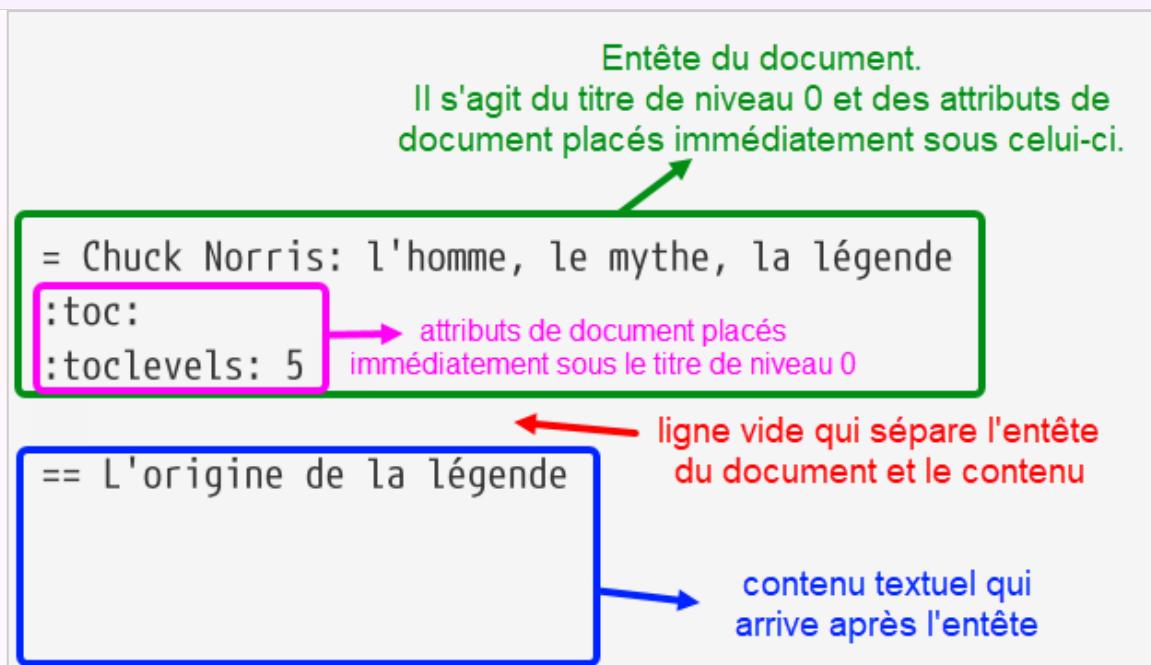


Figure 26. entête de document

L'entête de document AsciiDoc ne doit pas être confondu avec l'entête de page, c'est-à-dire le texte affiché dans la marge haute d'un fichier pdf.

#### 5.4.2. Personnaliser la table des matières

Il ne s'agit pas ici d'aborder la personnalisation de la mise en forme. Il s'agit de déterminer les niveaux de titre qui doivent être repris dans la table des matières et leur éventuelle numérotation entre autres choses.

Code AsciiDoc	Rendu
<h2 data-bbox="144 204 827 240">Spécifier la profondeur de la table des matières</h2> <p>L'attribut <code>:toclevels:</code> reçoit une valeur comprise entre <code>1</code> et <code>5</code>. Cette valeur fait référence aux niveaux de titre à afficher.</p> <p>La valeur <code>1</code> correspond au niveau de titre avec deux signes égal, la valeur <code>2</code> au niveau de titre avec trois signes égal, etc.</p> <p>C'est un <b>attribut de document</b> (puisque il agit sur le rendu final du document) et se doit d'être placé sous le titre principal ou immédiatement après un autre attribut de document.</p>	<h2 data-bbox="1033 204 1430 287">Chuck Norris: l'homme, le mythe, la légende</h2> <p>Table of Contents</p> <ul style="list-style-type: none"> <li><a href="#">L'origine de la légende</a></li> <li><a href="#">Enfance extraordinaire</a></li> <li><a href="#">Chuck Norris et les jouets</a></li> <li><a href="#">Les premières victoires</a></li> <li><a href="#">Les débuts de la force</a></li> <li><a href="#">Chuck Norris vs. la gravité</a></li> <li><a href="#">La naissance d'une légende</a></li> <li><a href="#">Pouvoirs surnaturels</a></li> <li><a href="#">Force absolue</a></li> <li><a href="#">Conclusion</a></li> </ul>
<pre data-bbox="171 698 838 887"> = Chuck Norris: l'homme, le mythe, la légende :toc: :toclevels: 3 ① ② == L'origine de la légende </pre>	<p>Figure 27. table des matières avec tous les titres rendus</p>  <p>N'hésitez pas à changer la valeur pour expérimenter le rendu.</p>
<p>① la table des matières doit afficher les titres jusqu'au niveau 3.</p> <p>L'attribut doit être placé immédiatement sous le titre principal ou sous un autre attribut de document ce qui est le cas ici avec l'attribut <code>:toc:</code> &lt;2&gt; ligne vide séparant le titre et ses attributs de document du reste du contenu.</p>	<h2 data-bbox="1033 1275 1399 1358">Chuck Norris: l'homme, le mythe, la légende</h2> <p>Table of Contents</p> <ul style="list-style-type: none"> <li><a href="#">1. L'origine de la légende</a> <ul style="list-style-type: none"> <li><a href="#">1.1. Enfance extraordinaire</a> <ul style="list-style-type: none"> <li><a href="#">1.1.1. Chuck Norris et les jouets</a></li> <li><a href="#">1.1.2. Les premières victoires</a></li> </ul> </li> <li><a href="#">1.2. Les débuts de la force</a> <ul style="list-style-type: none"> <li><a href="#">1.2.1. Chuck Norris vs. la gravité</a></li> <li><a href="#">1.2.2. La naissance d'une légende</a></li> </ul> </li> </ul> </li> </ul> <p>1. L'origine de la légende</p> <ul style="list-style-type: none"> <li><a href="#">1.1. Enfance extraordinaire</a> <ul style="list-style-type: none"> <li><a href="#">1.1.1. Chuck Norris et les jouets</a></li> <li><a href="#">1.1.2. Les premières victoires</a></li> </ul> </li> <li><a href="#">1.2. Les débuts de la force</a> <ul style="list-style-type: none"> <li><a href="#">1.2.1. Chuck Norris vs. la gravité</a></li> </ul> </li> </ul>

Code AsciiDoc	Rendu
<h2>Spécifier la profondeur des titres à numérotter</h2> <p>L'attribut de document <code>:sectnumlevels:</code> attend en valeur un niveau de compris entre <code>1</code> et <code>5</code>.</p> <pre>= Chuck Norris : l'homme, le mythe, la légende :toc: :toclevels: 2 :sectnums: :sectnumlevels: 2 ①  == L'origine de la légende</pre>	<p>Chuck Norris : l'homme, le mythe, la légende</p> <p>Table of Contents</p> <p>1. L'origine de la légende</p> <ul style="list-style-type: none"> <li>1.1. Enfance extraordinaire</li> <li>    Chuck Norris et les jouets</li> <li>    Les premières victoires</li> <li>1.2. Les débuts de la force</li> </ul> <p>1. L'origine de la légende</p> <p>1.1. Enfance extraordinaire</p> <p>    Chuck Norris et les jouets</p> <p>    Les premières victoires</p>
<p>① Les titres sont numérotés jusqu'aux titres de niveau 2 mais pas les autres.</p> <p>N'hésitez pas à changer la valeur pour expérimenter le rendu.</p> <pre>= Chuck Norris : l'homme, le mythe, la légende :toc: :toclevels: 5 :sectnums: :sectnumlevels: 2 :toc-title: Table des matières ①  == L'origine de la légende</pre> <p>① personnalisation du titre de la table des matières</p>	<p>Chuck Norris : l'homme, le mythe, la légende</p> <p>Table des matières</p> <p>1. L'origine de la légende</p> <ul style="list-style-type: none"> <li>1.1. Enfance extraordinaire</li> </ul> <p>Figure 29. profondeur de numérotation des titres</p> <p>Figure 30. La table des matières a pour titre « Table des matières »</p>

Si vous souhaitez aller plus loin avec la table des matières, voici des liens intéressants :

- [positionner la table des matières](#)
- [liste des attributs relatifs à la table des matières](#)

# 6. Les paragraphes

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 14:10 | Auteur : Emmanuel Ravrat

Durée de réalisation : 20min

Un **paragraphe** est un bloc de texte composé de plusieurs phrases liées entre elles par une idée commune.

Lorsqu'une nouvelle idée est abordée, un nouveau paragraphe doit être créé.

Il en va exactement de même avec AsciiDoc.

Code AsciiDoc	Rendu
<p><b>Ecrire un paragraphe</b></p> <p>Un paragraphe s'écrit au kilomètre en commençant bien contre la marge.</p> <div data-bbox="152 900 986 1051"><p>1 On raconte que Chuck Norris ne fait jamais de pompes, mais qu'il pousse simplement la Terre vers le bas. C'est dire s'il est puissant !</p></div>	<p>On raconte que Chuck Norris ne fait jamais de pompes, mais qu'il pousse simplement la Terre vers le bas. C'est dire s'il est puissant !</p>
<p><b>Faux retour à la ligne</b></p> <p>Un retour à la ligne dans le code n'est pas un retour à la ligne dans le rendu :</p> <div data-bbox="152 1320 946 1516"><p>1 Quand Chuck Norris écrit du code, les compilateurs demandent pardon pour les erreurs de syntaxe. 2 En fait, les ordinateurs n'ont même pas besoin de processeurs : ils fonctionnent uniquement grâce à la puissance de sa volonté.</p></div>	<p>Quand Chuck Norris écrit du code, les compilateurs demandent pardon pour les erreurs de syntaxe. En fait, les ordinateurs n'ont même pas besoin de processeurs : ils fonctionnent uniquement grâce à la puissance de sa volonté.</p>

Code AsciiDoc	Rendu
<h2>Un vrai retour à la ligne</h2> <p>Pour faire un <b>retour à la ligne</b>, il faut utiliser le caractère <code>+</code> à la fin de la ligne précédente :</p> <div data-bbox="171 422 944 653"><ol style="list-style-type: none"><li>1 Chuck Norris ne fait pas de sauvegardes. +</li><li>2 Les fichiers s'auto-duplicent par respect envers lui. +</li><li>3 Une fois, un virus a infecté son ordinateur. Quand Chuck a cliqué sur ce dernier pour l'exécuter, celui-ci s'est transformé en antivirus.</li></ol></div> <div data-bbox="184 772 247 833" data-label="Image"></div> <div data-bbox="296 738 975 864" data-label="Text"><p>Les retours à la ligne sont assez peu utilisés. Généralement, vous voudrez faire un nouveau paragraphe.</p></div>	<p>Chuck Norris ne fait pas de sauvegardes. Les fichiers s'auto-duplicent par respect envers lui. Une fois, un virus a infecté son ordinateur. Quand Chuck a cliqué sur ce dernier pour l'exécuter, celui-ci s'est transformé en antivirus.</p>
<h2>Faire un saut de ligne (nouveau paragraphe)</h2> <p>Pour créer un nouveau paragraphe, il faut laisser au moins une ligne vide :</p> <div data-bbox="171 1147 944 1379"><ol style="list-style-type: none"><li>1 Chuck Norris ne fait pas de sauvegardes. Les fichiers s'auto-duplicent par respect envers lui.</li><li>2</li><li>3 Une fois, un virus a infecté son ordinateur. Quand Chuck a cliqué sur ce dernier pour l'exécuter, celui-ci s'est transformé en antivirus.</li></ol></div>	<p>Chuck Norris ne fait pas de sauvegardes. Les fichiers s'auto-duplicent par respect envers lui. Une fois, un virus a infecté son ordinateur. Quand Chuck a cliqué sur ce dernier pour l'exécuter, celui-ci s'est transformé en antivirus.</p>

Code AsciiDoc	Rendu
<p><b>La bonne habitude à prendre</b></p> <p>Si un paragraphe est constitué de plusieurs phrases, <b>chaque phrase doit être écrite sur une nouvelle ligne</b>. Cela facilite la lecture du fichier et sa maintenance (ainsi que son versionnement que l'on verra bientôt)</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><p>1 Chuck Norris ne fait pas de sauvegardes. 2 Les fichiers s'auto-dupliquent par respect envers lui. 3 4 Une fois, un virus a infecté son ordinateur. 5 Quand Chuck a cliqué sur ce dernier pour l'exécuter, celui-ci s'est transformé en antivirus.</p></div>	<p>Chuck Norris ne fait pas de sauvegardes. Les fichiers s'auto-dupliquent par respect envers lui.</p> <p>Une fois, un virus a infecté son ordinateur. Quand Chuck a cliqué sur ce dernier pour l'exécuter, celui-ci s'est transformé en antivirus.</p>
<p><b>Paragraphe littéral</b></p> <p>Lorsqu'un paragraphe ne commence pas contre la marge, il est traité comme un <b>bloc littéral</b>, c'est-à-dire qu'il est rendu avec une police à espacement fixe et exactement comme il a été saisi (retours à la ligne inclus !).</p> <p>Voici un paragraphe qui commence par un espace :</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><p>Le paragraphe ci-après est précédé d'un espace ce qui a une signification pour AsciiDoc : le texte est traité comme un bloc de texte</p></div>	<p>Le paragraphe ci-après est précédé d'un espace ce qui a une signification pour AsciiDoc : le texte est traité comme un bloc de texte</p>

Code AsciiDoc	Rendu
<p><b>Le paragraphe de préambule</b></p> <p>Ce que j'appelle le <b>paragraphe de préambule</b> est le premier paragraphe écrit entre le <b>titre de niveau 0</b> et le <b>premier titre de niveau 1</b>.</p> <p>Il est rendu dans une police plus grande que celles des autres paragraphes.</p> <pre>= titre principal (niveau 0)  Ceci est un paragraphe de préambule. C'est un préambule au contenu qui va suivre. Il est écrit par défaut dans une police plus grande que celle des autres paragraphes.  Ceci est la suite du contenu. Il est écrit avec la taille de police de base.  == titre de niveau 1</pre> <p> Si le document n'a pas de titre de niveau 1, le premier paragraphe ne sera pas considéré comme un paragraphe de préambule.</p>	<p><b>titre principal (niveau 0)</b></p> <p>Ceci est un paragraphe de préambule. C'est un préambule au contenu qui va suivre. Il est écrit par défaut dans une police plus grande que celle des autres paragraphes.</p> <p>Ceci est la suite du contenu. Il est écrit avec la taille de police de base.</p> <p><b>préambule = 1er paragraphe placé entre le titre principal et le titre de niveau 1</b></p>

Pour en apprendre davantage sur les paragraphes, vous pouvez lire [la documentation](#).

## 7. Mise en forme du texte

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 14:15 | Auteur : Emmanuel Ravrat

Durée de réalisation : 15min

Les langages à balisage léger offrent assez peu de possibilités de mise en forme du texte. L'objectif est de pouvoir écrire du contenu sans perdre du temps à gérer celle-ci.

Cependant, ce n'est pas parce qu'il y a peu de possibilités qu'il n'y en a pas assez pour satisfaire 90% des besoins (et nous verrons comment couvrir les 10% restant plus tard). Le support que vous lisez est intégralement écrit en AsciiDoc et pourtant sa mise en forme n'a rien à envier à l'utilisation d'un logiciel de traitement de texte !

Quand on y pense, les besoins courants sont la mise en gras, en italique, le soulignement et le surlignement. Finalement, avez-vous besoin de plus de possibilités ? Que faites-vous de plus dans vos documents quand vous utilisez un traitement de texte tel que Word ou Writer ?

Pour commencer, voici les "balises" AsciiDoc qui permettent une mise en forme élémentaire :

Table 1. code AsciiDoc pour la mise en forme du texte

Code AsciiDoc	Rendu
voici du _texte en italique_	voici du <i>texte en italique</i>
Voici du *_texte en gras et italique_*	Voici du <b><i>texte en gras et italique</i></b>
Voici un mot #Surligné#	Voici un mot <b>Surligné</b>
Voici du contenu sur##ligné à l'intérieur d'u##ne chaîne	Voici du contenu <b>surligné</b> à l'intérieur d'une chaîne
Voici du `texte utilisant une font à espace constant`	Voici du <b>texte utilisant une font à espace constant</b>
voici du texte en `*_italique, gras et à espacement fixe_*`	voici du texte en <b>italique, gras et à espacement fixe</b>
Mettre en gras une lettre dans un mot : ho**s**pital	Mettre en gras une lettre dans un mot : <b>hospital</b>

Code AsciiDoc	Rendu
<pre>Utiliser l'espacement fixe da``ns une chaine de carac``tères</pre>	<p>Utiliser l'espacement fixe dans une chaine de carac`tères</p>
<p>Le caractère de formatage d'espacement fixe backtick ` doit être doublé lorsqu'il est utilisé dans un mot. Dans ce cas, la fin devra être marquée par un double backtick également</p>	
<pre>voici comment faire les exposants : 10^125^</pre>	<p>voici comment faire les exposants : 10<sup>125</sup></p>
<pre>voici comment exprimer une base 10~2~</pre>	<p>voici comment exprimer une base 10<sub>2</sub></p>

Si vous trouvez insuffisantes les possibilités de formatage du texte, vous pouvez lire les chapitres [chapitre 8, Mise en forme avec les rôles intégrés](#) et celui sur les [chapitre 27, Mise en forme à l'aide de rôles personnalisés](#).

## 8. Mise en forme avec les rôles intégrés

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 15:57 | Auteur : Emmanuel Ravrat

Durée de réalisation : 25min

Un rôle est un nom associé à un formatage de texte (écrire petit, écrire gros, souligner du texte, etc.).

AsciiDoc prévoit nativement des rôles, c'est-à-dire qu'ils sont intégrés au langage. C'est pour cela que l'on parle de rôles intégrés.

Il est possible d'appliquer un rôle à un paragraphe ou à un morceau de texte.

Pour appliquer un rôle à un paragraphe, il faut spécifier le nom du rôle précédé d'un point, entre crochets, sur la ligne qui précède le paragraphe :

```
1 [.big] ①
2 Ce texte est formaté avec le rôle 'big'.
3 Le texte est écrit plus gros.
4
5 Ce paragraphe est écrit dans sa taille de police de base.
6
7 [.small] ②
8 Ce texte est formaté avec le rôle 'small'.
9 Le texte est écrit plus petit.
10
11 [.text-right] ③
12 Ce texte est formaté avec le rôle 'text-right'.
13 Le texte est souligné.
```

① rôle **big** appliqué au paragraphe.

② rôle **small** appliqué au paragraphe.

③ rôle **text-right** appliqué au paragraphe.

Ce texte est formaté avec le rôle **big**. Le texte est écrit plus gros.

Ce paragraphe est écrit dans sa taille de police de base.

Ce texte est formaté avec le rôle **small**. Le texte est écrit plus petit.

Ce texte est formaté avec le rôle **underline**. Il est aligné à droite.

Figure 32. formatage de paragraphes à l'aide de rôle

Un rôle intégré peut être utilisé pour formater un morceau de texte. Dans ce cas, il faut encadrer le texte à formater avec le caractère # et précéder le morceau de texte par le nom du rôle précédé d'un point, entre crochets :

```
1 Voici ce [.big]#morceau de texte# formatté avec le rôle 'big'.
```

```

2
3 Voici ce [.small]#morceau de texte# formatté avec le rôle small 'small'.
4
5 Voici ce [.underline]#morceau de texte# formatté avec le rôle 'underline'.
6
7 Voici ce [.line-through]#morceau de texte# formatté avec le rôle 'line-through'.

```

Voici ce morceau de texte formatté avec le rôle **big**.

Voici ce morceau de texte formatté avec le rôle **small**.

Voici ce morceau de texte formatté avec le rôle **underline**.

Voici ce ~~morceau de texte~~ formatté avec le rôle **line-through**.

*Figure 33. formatage de morceaux de texte avec des rôles*

Voici les rôles les plus utiles sachant que certains d'entre eux ne sont pas utilisables sur un paragraphe **et** sur un morceau de texte :

rôle	formatage	para-graphhe	texte
<b>big</b>	Le texte est écrit dans une plus grande taille de police que la taille de base	✓	✓
<b>small</b>	Le texte est écrit dans une plus petite taille de police que la taille de base	✓	✓
<b>underline</b>	Le texte est souligné	✗	✓
<b>line-through</b>	Le texte est barré. C'est utile lorsqu'il faut montrer qu'un contenu a existé mais qu'il n'est plus valable.	✗	✓
<b>text-right</b>	Le texte est aligné à droite (mais pas à gauche).	✓	✗
<b>text-left</b>	Le texte est aligné à gauche.	✓	✗
<b>text-center</b>	Le texte est centré.	✓	✗

rôle	formatage	para-graphhe	texte
text-justify	Le texte est justifié.  Les lignes se terminent toutes contre la marge de droite (comme dans un livre). C'est le comportement par défaut.	✓	✗

Testez les différents rôles sur des paragraphes et des morceaux de texte afin de vous habituer à les utiliser. Si leur écriture vous paraît fastidieuse, vous verrez que grâce aux live templates, cela devient très rapide ! (bien plus rapide que de cliquer sur le bouton de mise en gras !)



Un nom de rôle ne peut pas contenir d'underscore `_`. Cela pourrait poser des problèmes dans l'application des styles de mise en forme.

Le nom de rôle fictif `tres_important` n'est pas valide alors que le nom `tres-important` est valide.

Il est possible de créer ses propres rôles (voir le chapitre [chapitre 27, Mise en forme à l'aide de rôles personnalisés](#))

Pour en savoir un peu plus sur les rôles intégrés, vous pouvez lire [la page de la documentation dédiée à cette notion](#).

# 9. Les listes

Version 1 | Dernière mise à jour : 26/08/2024 à 16:39 | Auteur : Emmanuel Ravrat

Durée de réalisation : 35min

## 9.1. Créer des listes non ordonnées

Une **liste non ordonnée** est une liste dont l'ordre des items est sans importance. Ils peuvent être énoncés dans n'importe quel ordre sans que cela change la compréhension du lecteur.

Code AsciiDoc	Rendu
<p>Ecrire une <b>liste non ordonnée</b></p> <p>Le caractère <code>*</code> ou <code>-</code> peut être utilisé pour lister les items d'une liste non ordonnée.</p> <p>Voici un exemple avec le caractère <code>*</code> :</p> <div style="border: 1px solid black; padding: 10px;"><p>* Quand Chuck Norris fait un push sur GitHub, les développeurs concurrents se retirent par respect. * Chuck Norris peut compiler du code en binaire avec sa pensée. * Chuck Norris ne fait pas de tests unitaires. Ses fonctions sont toujours parfaites du premier coup. * Chuck Norris peut écrire une boucle infinie... qui se termine. * Chuck Norris peut trouver la position du bit de poids fort en un seul regard. * Chuck Norris a appris à son chat à compiler du C++.</p></div>	<ul style="list-style-type: none"><li>• Quand Chuck Norris fait un push sur GitHub,...</li><li>• Chuck Norris peut compiler du code en binaire...</li><li>• Chuck Norris ne fait pas de tests unitaires. Ses fonctions sont toujours parfaites du premier...</li><li>• Chuck Norris peut écrire une boucle infinie...</li><li>• Chuck Norris peut trouver la position du bit...</li><li>• Chuck Norris a appris à son chat à compiler du C++.</li></ul>

Code AsciiDoc	Rendu
<p>Ecrire deux <b>listes séparées mais qui se suivent pas.</b></p> <p>Si vous avez besoin de faire deux listes qui se suivent, vous devez indiquer à AsciiDoc que les items de la seconde liste appartiennent à une nouvelle liste.</p> <p>L'utilisation d'un attribut de bloc vide <code>[]</code> permet d'indiquer qu'il s'agit d'une nouvelle liste :</p> <div style="border: 1px solid black; padding: 10px; margin-top: 10px;"><p>Voici deux listes séparées (l'espace est visible entre l'item 2 et l'item A) :</p><pre>* item 1 * item 2  [] ① * item A * item B</pre></div> <p>① La ligne contient un attribut de bloc mais vide. Cela permet de marquer le début de la nouvelle liste. Il doit être précédé d'une ligne vide.</p>	<p>Voici deux listes séparées (l'espace est visible entre l'item 2 et l'item A) :</p> <ul style="list-style-type: none"><li>• item 1</li><li>• item 2</li></ul> <ul style="list-style-type: none"><li>• item A</li><li>• item B</li></ul>

Code AsciiDoc	Rendu
<p>Ecrire une <b>liste imbriquée</b></p> <p>Il existe plusieurs façons d'écrire des listes imbriquées. Je vous donne celle que je juge la plus lisible.</p> <p>Pour créer un niveau supplémentaire d'imbrication, il suffit de doubler, tripler, etc. le caractère qui permet de créer un item de liste.</p> <p>Voici une liste avec un à deux niveaux d'imbrication.</p> <div style="border: 1px solid black; padding: 10px;"><ul style="list-style-type: none"><li>* Quand Chuck Norris fait un push sur GitHub, les développeurs concurrents se retirent par respect.</li><li>** Les commits de Chuck Norris sont si parfaits que même Linus Torvalds les accepte sans poser de questions.</li><li>** Chuck Norris n'a jamais eu besoin de faire un pull request. Le code se fusionne spontanément à son approche.</li> <li>* Chuck Norris peut trouver la position du bit de poids fort en un seul regard.</li><li>** Chuck Norris peut lire l'octet le moins significatif d'un octet.</li> <li>* Chuck Norris a appris à son chat à compiler du C++.</li><li>** Le chat de Chuck Norris est un contributeur GitHub populaire.</li><li>*** Le chat de Chuck Norris utilise le Dark Mode avec du texte dans la même couleur que le fond</li><li>*** Le chat de Chuck Norris n'utilise jamais de souris pour coder, il l'a mangée</li><li>** Les vidéos de tutoriels de programmation de son chat sont meilleures que celles de votre professeur.</li></ul></div> <p> Je vous conseille de sauter une ligne entre les items principaux afin de faciliter la lecture. Cela n'impacte pas le rendu.</p>	<ul style="list-style-type: none"><li>• Quand Chuck Norris fait un push sur GitHub,...<ul style="list-style-type: none"><li>◦ Les commits de Chuck Norris sont si parfaits...</li><li>◦ Chuck Norris n'a jamais eu besoin de faire...</li></ul></li><li>• Chuck Norris peut trouver la position du bit...<ul style="list-style-type: none"><li>◦ Chuck Norris peut lire l'octet le moins significatif d'un octet.</li></ul></li><li>• Chuck Norris a appris à son chat à compiler...<ul style="list-style-type: none"><li>◦ Le chat de Chuck Norris est un contributeur...<ul style="list-style-type: none"><li>▪ Le chat de Chuck Norris utilise le Dark Mode...</li><li>▪ Le chat de Chuck Norris n'utilise jamais de...</li></ul></li><li>◦ Les vidéos de tutoriels de programmation de...</li></ul></li></ul>

Code AsciiDoc	Rendu
<p>Une <b>liste avec une image</b></p> <p>Un item de liste peut contenir absolument n'importe quoi (un paragraphe, un tableau, un lien, un bloc de code, une image, une <a href="#">liste ordonnée</a> etc.).</p> <p>Cependant, une image est affichée sur une nouvelle ligne. Celle-ci apparaît au même niveau que la puce à laquelle elle est rattachée alors qu'elle devrait être alignée sur le début du texte de cette puce.</p> <p>Pour éviter cet écueil, il faut utiliser le caractère <code>+</code> qui est un caractère de rattachement de l'image à l'item de la liste. L'image est ainsi alignée au même niveau que le contenu textuel.</p> <p>Voici un exemple avec une image de notre bien aimé Chuck ! (ne lappelez surtout pas Chucky !)</p> <pre data-bbox="147 945 986 1282"> * Chuck Norris a usé plus de 1500 tétines + ① .les tétines de Chuck Norris image::images/tetine_chuck_norris.png[]  * Chuck Norris avait toutes ses dents d'adulte à 8 mois </pre> <p>① Le caractère <code>+</code> permet d'indiquer que l'élément qui suit doit être rattaché à l'item principal. C'est obligatoire pour les éléments qui font un bloc comme les images.</p> <p>Essayez en retirant le caractère <code>+</code> pour comprendre la différence visuelle.</p>	<ul style="list-style-type: none"> <li>• Chuck Norris a usé plus de 1500 tétines</li> </ul>   <p><i>Figure 34. les tétines de Chuck Norris</i></p>

Si vous voulez en savoir plus sur les listes non ordonnées, voici quelques liens :

- [liste non ordonnées](#)
- [listes complexes](#)
- [attacher un élément à un ancêtre](#) alors que l'on est dans un sous-item.
- [liste de description](#)

## 9.2. Créer des listes ordonnées

Une **liste ordonnée** est une liste d'items dont l'ordre dans lequel ils sont énoncés est important. Par exemple, les étapes d'une recette de cuisine doivent être ordonnées (on ne commence pas à mettre

le moule du gâteau au four avant d'avoir préparé le gâteau !).

Pour créer un item numéroté, il suffit d'utiliser le caractère point **.** suivi d'un espace.

Code AsciiDoc	Rendu
<p><b>Liste numérotée</b></p> <p>Voici l'ordre des étapes à respecter pour espérer pouvoir passer 5 minutes en présence de Chuck Norris :</p> <ul style="list-style-type: none"><li>. trouver une licorne dorée qui a mangé une portion de poussière de lune.</li><li>. apprendre le langage secret des écureuils</li><li>. développer un algorithme qui calcule le nombre exact de cheveux sur la tête de Chuck Norris sans qu'il le sache.</li></ul>	<p>Voici l'ordre des étapes à respecter pour espérer pouvoir passer 5 minutes en présence de Chuck Norris :</p> <ol style="list-style-type: none"><li>1. trouver une licorne dorée qui a mangé une portion de poussière de lune.</li><li>2. apprendre le langage secret des écureuils</li><li>3. développer un algorithme qui calcule le nombre exact de cheveux sur la tête de Chuck Norris sans qu'il le sache.</li></ol>

Code AsciiDoc	Rendu
<p><b>Liste ordonnée avec des lettres</b></p> <p>En précédent la liste du nom du style <code>loweralpha</code>, vous aurez des lettres minuscules à la place des numéros :</p> <div data-bbox="165 422 901 534" style="border: 1px solid #ccc; padding: 10px;"><p>Voici l'ordre des étapes à respecter pour espérer pouvoir passer 5 minutes en présence de Chuck Norris :</p></div> <p>[loweralpha] ①</p> <ul style="list-style-type: none"><li>. trouver une licorne dorée qui a mangé une portion de poussière de lune.</li><li>. apprendre le langage secret des écureuils</li><li>. développer un algorithme qui calcule le nombre exact de cheveux sur la tête de Chuck Norris sans qu'il le sache.</li></ul> <p>① le nom de style <code>loweralpha</code> (entre crochets) indique que des lettres minuscules doivent être utilisées.</p> <div data-bbox="187 1170 250 1230" style="background-color: #0070C0; width: 40px; height: 30px; display: flex; align-items: center; justify-content: center; margin-bottom: 10px;"></div> <div data-bbox="298 1037 979 1201" style="background-color: #E0F2F1; padding: 10px;"><p>Voici d'autres styles prédéfinis qui peuvent être utilisés: <code>loweralpha</code>, <code>uperalpha</code>, <code>lowerroman</code>, <code>upperroman</code>, <code>lowergreek</code>, <code>decimal</code>, <code>arabic</code> (valeur par défaut).</p></div> <div data-bbox="298 1239 979 1363" style="background-color: #E0F2F1; padding: 10px;"><p>Il est même possible de créer son propre style (voir le chapitre : <a href="#">chapitre 27, Mise en forme à l'aide de rôles personnalisés</a>)</p></div>	<p>Voici l'ordre des étapes à respecter pour espérer pouvoir passer 5 minutes en présence de Chuck Norris :</p> <ul style="list-style-type: none"><li>a. trouver une licorne dorée qui a mangé une portion de poussière de lune.</li><li>b. apprendre le langage secret des écureuils</li><li>c. développer un algorithme qui calcule le nombre exact de cheveux sur la tête de Chuck Norris sans qu'il le sache.</li></ul>

Code AsciiDoc	Rendu
<h3 data-bbox="144 204 854 240">Modifier le premier numéro d'une liste ordonnée</h3> <p data-bbox="144 280 1006 440">Il est possible de contrôler la valeur du premier numéro d'une liste ordonnée (que celle-ci soit numérique ou non). L'attribut à utiliser est <code>start</code> et peut être utilisé sans ou en complément d'autres attributs :</p> <div data-bbox="173 505 946 579" style="border: 1px solid black; padding: 10px;"> <p data-bbox="173 505 946 579">Voici une liste des choses à faire pour passer 5 min avec Chuck :</p> </div> <pre data-bbox="173 624 933 893" style="border: 1px solid black; padding: 10px;"> [loweralpha,start=6] ① . trouver une licorne dorée qui a mangé une portion de poussière de lune. . apprendre le langage secret des écureuils . développer un algorithme qui calcule le nombre exact de cheveux sur la tête de Chuck Norris sans qu'il le sache. </pre> <p data-bbox="152 945 1006 1019">① Avec <code>start=6</code>, la liste commencera à la 6ème valeur, soit la lettre <code>f</code> puisque la liste est stylé avec <code>loweralpha</code>.</p>	<p data-bbox="1030 204 1430 318">Voici une liste des choses à faire pour passer 5 min avec Chuck :</p> <ul style="list-style-type: none"> <li data-bbox="1044 361 1430 476">f. trouver une licorne dorée qui a mangé une portion de poussière de lune.</li> <li data-bbox="1044 505 1430 579">g. apprendre le langage secret des écureuils</li> <li data-bbox="1044 608 1430 804">h. développer un algorithme qui calcule le nombre exact de cheveux sur la tête de Chuck Norris sans qu'il le sache.</li> </ul>
<h3 data-bbox="144 1073 859 1109">liste ordonnée imbriquée dans une liste ordonnée</h3> <p data-bbox="144 1149 1006 1264">Pour imbriquer une liste ordonnée dans une liste ordonnée, il suffit de doubler, tripler, etc., le caractère <code>.</code> en fonction du niveau d'imbrication.</p> <div data-bbox="173 1329 946 1403" style="border: 1px solid black; padding: 10px;"> <p data-bbox="173 1329 946 1403">Voici une liste des choses à faire pour passer 5 min avec Chuck :</p> </div> <pre data-bbox="173 1448 946 1758" style="border: 1px solid black; padding: 10px;"> . trouver une licorne dorée qui a mangé une portion de poussière de lune. .. la licorne doit avoir plus de 450 ans .. la poussière de lune doit venir de sa face cachée . apprendre le langage secret des écureuils .. savoir raconter une blague de gland ... avoir un chêne autour du cou et à son vélo ... ramasser 40kg de noisettes </pre>	<p data-bbox="1030 1073 1430 1188">Voici une liste des choses à faire pour passer 5 min avec Chuck :</p> <ol style="list-style-type: none"> <li data-bbox="1044 1230 1430 1367">1. trouver une licorne dorée qui a mangé une portion de poussière de lune.       <ol style="list-style-type: none"> <li data-bbox="1092 1374 1430 1448">a. la licorne doit avoir plus de 450 ans</li> <li data-bbox="1092 1462 1430 1590">b. la poussière de lune doit venir de sa face cachée</li> </ol> </li> <li data-bbox="1044 1619 1430 1693">2. apprendre le langage secret des écureuils       <ol style="list-style-type: none"> <li data-bbox="1092 1709 1430 1783">a. savoir raconter une blague de gland           <ol style="list-style-type: none"> <li data-bbox="1156 1799 1430 1927">i. avoir un chêne autour du cou et à son vélo</li> <li data-bbox="1156 1940 1430 2014">ii. ramasser 40kg de noisettes</li> </ol> </li> </ol> </li> </ol>

Code AsciiDoc	Rendu
<p><b>liste non ordonnée imbriquée dans une liste ordonnée</b> Une liste peut contenir une liste non ordonnée :</p> <pre data-bbox="165 345 933 579"> . trouver une licorne dorée qui a mangé une portion de poussière de lune. * la licorne doit avoir plus de 450 ans * la poussière de lune doit venir de sa face cachée . apprendre le langage secret des écureuils * savoir raconter une blague de gland </pre>	<ol style="list-style-type: none"> <li>1. trouver une licorne dorée qui a mangé une portion de poussière de lune. <ul style="list-style-type: none"> <li>◦ la licorne doit avoir plus de 450 ans</li> <li>◦ la poussière de lune doit venir de sa face cachée</li> </ul> </li> <li>2. apprendre le langage secret des écureuils <ul style="list-style-type: none"> <li>◦ savoir raconter une blague de gland</li> </ul> </li> </ol>

Si vous voulez en savoir plus sur les listes ordonnées, voici quelques liens :

- [liste ordonnées](#)
- [listes complexes](#)
- [attacher un élément à un ancêtre](#) alors que l'on est dans un sous-item.
- [liste de description](#)

## 9.3. Créer des checklists

Une **checklist** est une liste non ordonnée dont les items peuvent être marqués comme validés.

Code AsciiDoc	Rendu
<p>Une case à cocher est créée avec des crochets. Si la case n'est pas cochée, il faut un espace entre les crochets. Si la case doit être cochée, il faut utiliser soit un <code>*</code>, soit un <code>x</code> :</p> <pre data-bbox="165 1603 959 1971"> Voici les choses à faire pour espérer avoir un regard de Chuck Norris :  * [ ] trouver une licorne dorée qui a mangé une portion de poussière de lune. * [x] apprendre le langage secret des écureuils * [*] développer un algorithme qui calcule le nombre exact de cheveux sur la tête de Chuck Norris sans qu'il le sache. </pre>	<p>Voici les choses à faire pour espérer avoir un regard de Chuck Norris :</p> <ul style="list-style-type: none"> <li><input type="checkbox"/> trouver une licorne dorée qui a mangé une portion de poussière de lune.</li> <li><input checked="" type="checkbox"/> apprendre le langage secret des écureuils</li> <li><input checked="" type="checkbox"/> développer un algorithme qui calcule le nombre exact de cheveux sur la tête de Chuck Norris sans qu'il le sache.</li> </ul>

# 10. Les images

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 17:13 | Auteur : Emmanuel Ravrat

Durée de réalisation : 35min

## 10.1. Copier-coller une image

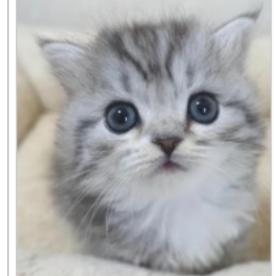
Vous avez deux possibilités pour ajouter une image à votre document :

- vous collez manuellement le fichier de l'image dans un dossier puis vous indiquez dans le document AsciiDoc le chemin vers celle-ci
- vous copiez l'image dans le presse-papier **CTRL + C** et vous la collez directement dans l'IDE via un **CTRL + V**. Dans ce cas, l'IDE (grâce au plugin AsciiDoc) ouvre une boîte de dialogue afin que vous déterminiez le dossier de destination et le nom du fichier. C'est le plugin qui écrit pour vous le chemin vers l'image.

 L'IDE est capable de gérer le « collage » de l'image depuis le presse-papier grâce au plugin AsciiDoc. Je rappelle que j'utilise IntelliJ Community (gratuit) ou n'importe quel IDE de JetBrains dans lequel j'installe le plugin AsciiDoc (voir le cours sur le choix et l'installation d'un IDE).

## 10.2. Insertion et fonctionnalités concernant les images

Code AsciiDoc	Rendu
<p>Insérer une image située dans le dossier <code>images</code>. Le chemin est relatif au fichier AsciiDoc.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"><code>image::images/chat_mignon.png[] ① ②</code></div> <p>① La macro <code>image::</code> est bien constituée du double caractère double points `::`.</p> <p>② L'image est placée dans le dossier <code>images</code> créé dans le même répertoire que le fichier AsciiDoc.</p>	
<p> Une image occupe par défaut l'intégralité de l'espace horizontal même si ses dimensions n'occupent pas tout cet espace. C'est-à-dire qu'il ne peut pas y avoir d'autres éléments à gauche ou à droite de l'image. L'image est un <b>bloc</b> comme un paragraphe peut l'être.</p>	

Code AsciiDoc	Rendu
<h3>Réduire une image</h3> <p>La largeur peut être spécifiée en % ou en pixel. Dans ce dernier cas, pas besoin de préciser une unité. La hauteur peut être spécifiée à la place ou en plus de la largeur avec <code>height</code></p> <pre data-bbox="176 467 763 503">image::images/chat_mignon.png[width=40%]</pre>	
<h3>Image avec un titre / une légende</h3> <pre data-bbox="176 680 763 759">.image de chat trop mignon ① image::images/chat_mignon.png[width=40%]</pre> <p>① il n'y a pas d'espace après le point et la première lettre du titre</p> <p>Le titre de l'image contient une étiquette de titre avec le texte « Figure » suivi d'un numéro d'apparition de l'image au sein du document. C'est le comportement par défaut.</p>	 <p>Figure 35. image de chat trop mignon</p>
<h3>Personnalisation ou suppression de l'étiquette de titre</h3> <p>La personnalisation de l'étiquette de titre qui par défaut est « Figure N » se fait via l'attribut de document <code>figure-caption</code></p> <pre data-bbox="176 1309 636 1388">= titre principal du document :figure-caption: Image perso ①</pre> <p>① attribut indiquant l'étiquette à utiliser pour toutes les images</p>	 <p>Image perso 2. image de chat trop mignon</p>
<h3>Image avec texte alternatif (utile si l'image n'est pas trouvée)</h3> <pre data-bbox="176 1657 886 1736">image::images/chat_licorne.png["Un chat avec une corne"]</pre>	<p>[Un chat avec une corne]   images/chat_licorne.png</p> <p>Figure 36. texte alternatif utilisé lorsque l'image n'est pas trouvée</p>

Code AsciiDoc	Rendu
<p><b>Image insérée dans une ligne de texte</b></p> <div data-bbox="171 309 941 386"><p>Ce petit chat image:images/chat_mignon.png[width=8%] est trop mignon ①</p></div>	<p>Ce petit chat  est trop mignon.</p>
<p>① image insérée dans une ligne de texte avec la macro <code>image</code> suivi d'un seul caractère :</p> <p>une image insérée dans une ligne nécessite d'utiliser la macro <code>image</code> avec un seul caractère :.</p> <p>Ceci est la macro pour insérer une image de type "bloc", c'est-à-dire qui occupe l'intégralité de l'espace horizontal.</p> <div data-bbox="330 871 470 907"><p><code>image::[]</code></p></div> <p>Ceci est la macro pour insérer une image de type "en ligne", c'est-à-dire qui occupe que l'espace relatif à sa taille. Il n'y a qu'un seul caractère :</p> <div data-bbox="330 1154 454 1190"><p><code>image:[]</code></p></div>	
<p><b>Image issue d'une URL</b></p> <p>Pour utiliser une image issue d'une URL, il suffit d'utiliser le lien vers cette image à la place du chemin habituel de l'image.</p> <div data-bbox="171 1507 959 1709"><p><code>image::https://cdn.wamiz.fr/cdn-cgi/image/format=auto,quality=80,width=664,height=373.5,fit=cover/article/main-picture/2017chatonsmignons20rect-fb-58e4c57735850.jpg[width=40%]</code></p></div> <div data-bbox="187 1843 250 1904"></div> <div data-bbox="298 1785 975 1949"><p>La technique d'utilisation d'une image à partir d'une URL n'est pas prise en charge dans le fichier pdf qui sera généré à partir du fichier AsciiDoc.</p></div>	<p>[2017chatonsmignons20rect fb-58e4c57735850]   <a href="https://cdn.wamiz.fr/cdn-cgi/image/format=auto,quality=80,width=664,height=373.5,fit=cover/article/main-picture/2017chatonsmignons20rect-fb-58e4c57735850.jpg">https://cdn.wamiz.fr/cdn-cgi/image/format=auto,quality=80,width=664,height=373.5,fit=cover/article/main-picture/2017chatonsmignons20rect-fb-58e4c57735850.jpg</a></p> <p>Le rendu pdf ne permet pas de charger l'image (mais cette possibilité fonctionne très bien avec la sortie HTML)</p>

Code AsciiDoc	Rendu
<p><b>Image qui sert de lien cliquable</b></p> <p>Le lien est placé entre chevrons sur la ligne qui précède l'image :</p> <div data-bbox="165 422 873 500" style="border: 1px solid #ccc; padding: 10px;"><p>En cliquant sur ce petit chat, j'arrive sur une recherche Google sur les chats trop mignons.</p></div> <div data-bbox="165 541 959 662" style="border: 1px solid #ccc; padding: 10px;"><pre>[link=https://www.google.com/search?q=chat+trop+mignon] ① image::images/chat_mignon.png[width=40%]</pre></div> <p>① lien cliquable attaché à l'image. Ce lien n'est pas visible mais bien présent.</p> <p>Le lien peut être écrit directement dans les attributs de l'image comme pour le cas de l'image en ligne ci-après, mais je trouve cela moins pratique et moins lisible).</p> <div data-bbox="298 997 975 1075" style="border: 1px solid #ccc; padding: 10px;"><p>Une image "en ligne" peut être cliquable. Dans ce cas, le lien est à placer dans les crochets tel que :</p></div> <div data-bbox="182 1149 250 1212" style="background-color: #0070C0; width: 40px; height: 28px; display: inline-block;"></div> <div data-bbox="330 1143 941 1338" style="border: 1px solid #ccc; padding: 10px;"><p>1 Cliquez sur ce chat mignon image:images/chat_mignon.png[width=8%, link=https://www.google.com/search?q=chat+trop+mignon] pour lancer une recherche Google.</p></div>	<p>En cliquant sur ce petit chat, j'arrive sur une recherche Google sur les chats trop mignons.</p>  <p>Cliquez sur ce chat mignon  pour lancer une recherche Google.</p>

 Le chemin d'une image est toujours relatif au fichier depuis lequel la génération du document final (pdf, html, etc.) est réalisée. Cela signifie que AsciiDoc s'attend à trouver les images au même niveau que le fichier AsciiDoc.

C'est une information importante sur laquelle nous reviendrons lorsque nous aborderons les documents composés de plusieurs fichiers AsciiDoc.

Si vous devez remplacer une image par une autre, vous avez deux possibilités :

- vous copiez l'image dans le presse papier puis vous la collez dans votre fichier (là où vous écrivez). Via la fenêtre qui s'affiche, vous choisissez l'image à remplacer et vous validez. Le plugin va vous demander si vous souhaitez remplacer l'image existante. Vous répondez oui.
- vous pouvez remplacer le fichier dans le dossier des images en veillant à lui donner le même nom que l'image remplacée.

Pour supprimer une image, supprimez d'abord l'image du dossier image puis supprimer la ligne AsciiDoc qui fait le lien vers l'image. Le faire dans cet ordre présente l'avantage de vérifier le nom de l'image à supprimer en cas d'oubli de celui-ci.

## 10.3. Positionner une image

Une image peut être utilisée avec des attributs de positionnement.

### Positionnement par défaut (à gauche)

Par défaut, une image est positionnée contre la marge de gauche :

```
.image positionnée par défaut à gauche  
image::images/chat_mignon.png[width=20%]  
  
//c'est équivalent à  
image::images/chat_mignon.png[width=20%, align="left"]
```



Figure 2. image positionnée par défaut à gauche

### Positionnement à droite

L'attribut `align` avec la valeur `right` permet un alignement à droite.

```
.image positionnée par défaut à gauche  
image::images/chat_mignon.png[width=20%,align="right"]
```



Figure 2. image positionnée par défaut à droite

Vous aurez remarqué que la légende de l'image reste positionnée à gauche. Nous reviendrons sur ce point un peu plus loin.

### Positionnement au centre

L'attribut `align` avec la valeur `center` permet un alignement à droite.

```
.image positionnée par défaut à gauche  
image::images/chat_mignon.png[width=20%,align="center"]
```



Figure 3. image positionnée par défaut à gauche

Vous aurez remarqué que la légende de l'image reste positionnée à gauche. Nous reviendrons sur ce point un peu plus loin.

La légende peut être positionnée automatiquement en fonction de l'alignement de l'image. Cela nécessite d'intervenir dans le fichier de thème (voir le chapitre [chapitre 26, Le fichier de thème PDF](#)).

Pour les plus impatients qui ont déjà un fichier de thème personnalisé qui étend le fichier de thème par défaut, vous pouvez indiquer que l'alignement d'une légende (*caption*) doit hériter de l'alignement de l'image à laquelle elle se rattache :



```
//dans le fichier de thème  
extends: default  
# ...  
image:  
caption:  
align: inherit
```

Si vous voulez en savoir plus sur le positionnement, notamment pour le format de sortie HTML, vous pouvez lire la [page de la documentation dédiée au positionnement des images](#).

# 11. Les liens

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 17:24 | Auteur : Emmanuel Ravrat

Durée de réalisation : 25min

## 11.1. Les liens web



Tous les liens contenus dans un fichier AsciiDoc peuvent être ouverts en maintenant la touche **CTRL** enfoncee etc en cliquant sur le lien.

Code AsciiDoc	Rendu
<p><b>Créer un lien hypertexte automatique</b></p> <p>AsciiDoc est capable de détecter une URL et la transforme automatiquement en lien cliquable.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><p>Voici les résultats d'une recherche « chuck norris vs superman » :</p><p><a href="https://www.google.com/search?q=chuck+norris+vs+superman">https://www.google.com/search?q=chuck+norris+vs+superman</a></p></div>	<p>Voici les résultats d'une recherche « chuck norris vs superman » :</p> <p><a href="https://www.google.com/search?q=chuck+norris+vs+superman">https://www.google.com/search?q=chuck+norris+vs+superman</a></p>
<div style="display: flex; align-items: center;"> Il faut que le protocole soit présent dans le lien. Ici, il s'agit du protocole <b>https</b>, mais ce peut être les protocoles <b>http, ftp, mailto, irc</b>.</div>	
<p><b>Créer un lien avec texte personnalisé</b></p> <p>Pour personnaliser le texte cliquable, il suffit d'ajouter une paire de crochets au bout du lien avec le texte personnalisé :</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><p>Voici les résultats d'une recherche «</p><p><a href="https://www.google.com/search?q=chuck+norris+vs+superman[chuck norris vs superman]">https://www.google.com/search?q=chuck+norris+vs+superman[chuck norris vs superman]</a> »</p></div>	<p>Voici les résultats d'une recherche « <b>chuck norris vs superman</b> »</p>

Code AsciiDoc	Rendu
<p><b>Ecrire un lien qui ne doit pas être actif.</b></p> <p>Un lien désactivé est un texte détecté comme étant un lien alors que vous ne souhaitez pas qu'il soit un lien.</p> <p>Pour désactiver le lien, il suffit d'utiliser un antislash en début du lien :</p> <div data-bbox="165 541 959 698" style="border: 1px solid #ccc; padding: 10px;"><p>Voici les résultats d'une recherche « chuck norris vs superman » : \https://www.google.com/search?q=chuck+norris+vs+superman</p></div>	<p>Voici les résultats d'une recherche « chuck norris vs superman » : <a href="https://www.google.com/search?q=chuck+norris+vs+superman">https://www.google.com/search?q=chuck+norris+vs+superman</a></p>

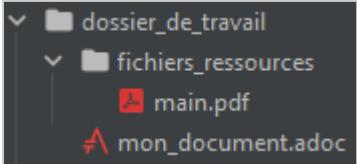
Si vous avez des besoins non couverts par ce chapitre, vous pouvez lire la documentation sur les points suivants :

- la [documentation sur les liens](#).
- [les macros URL](#)

## 11.2. Les liens vers des fichiers



Les liens vers des fichiers peuvent être ouverts depuis la fenêtre d'édition du fichier AsciiDoc en appuyant sur la touche **CTRL** tout en cliquant sur le lien.

AsciiDoc	Rendu
<h2>Créer un lien relatif vers un fichier</h2> <p>Un lien relatif signifie que le lien est exprimé en partant du fichier depuis lequel le document final sera rendu vers le fichier à lier.</p> <p>L'avantage d'un lien relatif est que vous restez indépendant du système depuis lequel vous éditez votre fichier.</p> <p>Imaginez l'arborescence suivante :</p>  <pre>└── dossier_de_travail     └── fichiers_ressources         ├── main.pdf         └── mon_document.adoc</pre>	<p>Pour ouvrir le fichier, cliquez sur ce lien : <a href="#">fichiers_ressources/main.pdf</a></p>
<p>Pour faire un lien depuis le fichier AsciiDoc <code>mon_document.adoc</code> vers le fichier pdf <code>main.pdf</code>, vous devez utiliser la macro <code>link</code> tel que <a href="#">chemin/relatif/vers/le_fichier</a></p> <div data-bbox="171 1006 838 1087" style="border: 1px solid #ccc; padding: 10px;"><p>Pour ouvrir le fichier, cliquez sur ce lien : <code>link:fichiers_ressources/main.pdf[]</code></p></div>	
<h2>Personnaliser le texte du lien vers un fichier</h2> <p>Personnaliser le texte d'un lien vers un fichier est identique à la <a href="#">personnalisation du lien URL</a>. Le texte personnalisé doit être écrit entre les crochets de la macro <code>link</code> :</p> <div data-bbox="171 1417 813 1498" style="border: 1px solid #ccc; padding: 10px;"><p>Pour ouvrir le fichier, cliquez sur ce lien <code>link:fichiers_ressources/main.pdf[]</code></p></div>	<p>Pour ouvrir le fichier, cliquez sur ce lien</p>

AsciiDoc	Rendu
<p><b>Cas d'un lien avec espaces vers un fichier</b></p> <p>Si vous nommez vos dossiers et fichiers avec des espaces (ce que je déconseille fortement), la macro <code>link</code> sera traitée comme du texte simple.</p> <p>Dans l'exemple ci-après, le dossier <code>fichiers ressources</code> contient un espace. La macro est neutralisée.</p> <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"><p>Pour ouvrir le fichier, cliquez sur ce lien <code>link:fichiers ressources/main.pdf[]</code></p></div> <p>Dans ce cas, vous devez encadrer le lien avec la macro <code>pass:[]</code> :</p> <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"><p>Pour ouvrir le fichier, cliquez sur ce lien <code>link:pass[fichiers ressources/main.pdf]</code></p></div>	<ul style="list-style-type: none"><li>Utilisation d'un lien contenant un espace dans le chemin relatif : « Pour ouvrir le fichier, cliquez sur ce lien <code>link:fichiers ressources/main.pdf[]</code> »</li><li>Utilisation des crochets autour du lien relatif : « Pour ouvrir le fichier, cliquez sur ce lien <code>fichiers ressources/main.pdf</code> »</li></ul>

	<p>Il est possible d'utiliser un chemin absolu pour cibler le fichier mais c'est une pratique que je déconseille. Dès lors que vous changez de machine, votre chemin ne sera plus valide.</p> <p>Un <b>chemin asbolu</b> est un chemin exprimé depuis la racine du disque sur lequel est stocké le fichier.</p>
---	---

Pour des cas plus complexes et d'autres fonctionnalités, vous pouvez lire la documentation :

- [les macros de lien](#)
- [les URL complexes](#)
- [les attributs de macros de lien et d'URL](#)
- [liste des attributs utilisables dans les macros de lien et d'URL](#)
- <https://docs.asciidoctor.org/asciidoc/latest/macros/xref-text-and-style/>

# 12. Utiliser des icônes

Version 1 | Dernière mise à jour : 26/08/2024 à 17:50 | Auteur : Emmanuel Ravrat

Durée de réalisation : 20min

Si vous souhaitez insérer des icônes dans un document AsciiDoc, il faut utiliser la macro en ligne `icon:[]` de la façon suivante :

```
icon:nom-de-l-icone[]
```

Les icônes qui peuvent être utilisées sont les icônes issues de Font Awesome :

- Voir les icônes de [Font Awesome 5](#)



L'utilisation d'icônes de Font Awesome 4 est dépréciée.

Les icônes de Font Awesome sont incluses par défaut dans Asciidoctor (l'outil qui transforme votre fichier AsciiDoc en fichier pdf). Vous n'avez donc rien à faire de particulier pour en profiter.

Pour insérer une icône, il faut utiliser la macro `icon:[]`.

```
1 icon:nom-de-l-icone[]
```

Pour savoir quel nom d'icône utiliser, rendez-vous sur [la page des icônes de Font Awesome 5](#). Une fois l'icône de votre choix repérée, cliquez-dessus. Vous voyez son nom et son "style"

Pour savoir quel nom d'icône utiliser, cliquez sur une des icônes libres de Font Awesome 5 puis repérer `fas` ou `far` ainsi que le nom de l'icône :

The screenshot shows the Font Awesome 5 icon library interface. At the top, there's a search bar with the text "wine-bottle" and a red border. To the right of the search bar, the text "nom de l'icône à utiliser" is displayed in red. Below the search bar, there are two small icons: "f72f" and a wine bottle icon. A horizontal scroll bar is visible below these elements. At the bottom, there's a dark navigation bar with tabs for "HTML", "REACT", "VUE", and "SVG". The "HTML" tab is selected. In the "HTML" tab, there is a code snippet: `<i class="fas fa-wine-bottle"></i>`. A red arrow points from this code snippet to a callout box on the right. The callout box contains the text "style de l'icône : fas = solid pour une icône pleine, far = regular pour une icône vide," written in red.

`icon:fas-wine-bottle[]` ce qui donne



A ce jour (26/08/2024), les icônes ne sont pas rendues dans la fenêtre de prévisualisation de l'ide.



A ce jour (26/08/2024), une icône n'est pas toujours rendue dans les deux styles `regular` et `solid` dans le fichier pdf. Pensez à vérifier leur rendu.

Voici un tableau qui illustre l'utilisation de la même icône dans deux styles différents :

Table 2. Exemple de rendu d'icône issue de Font Awesome 5

AsciiDoc	icone attendue	icone prévisualisée dans l'éditeur	icone dans le pdf
<code>icon:fas-heart[]</code> le <code>s</code> de <code>fas</code> indique "solid"	✓	aucune ✕	✓
<code>icon:far-heart[]</code> le <code>r</code> de <code>far</code> indique "regular"	✓	aucune ✕	✓

Il est possible d'agir sur le rendu de l'icône en utilisant des attributs. Ces attributs sont à préciser entre les crochets.

Voici des icônes qui font X fois leur taille de base grâce à l'attribut `size` :

```
icon:fas-grin-tongue-wink[4x] ①
icon:far-grin-tongue-wink[size=2x] ②
icon:far-grin-tongue-wink[] ③
```

① Le premier attribut est la taille, donc si aucun nom d'attribut n'est précisé, c'est qu'il s'agit de la taille

② l'attribut de taille est explicitement mentionné

③ La taille n'est pas spécifiée. C'est équivalent à `size=1x`, c'est-à-dire la taille de police courante.

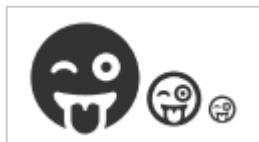


Figure 37. icônes dont la taille est modifiée avec l'attribut `size`

En plus de la taille, un rôle peut être ajouté pour formater l'icône (couleur, taille, fond).

Voici des icônes formatées avec un rôle :

```
icon:fas-file[role=filename]  
icon:fas-file[role=big-green]  
icon:fas-file[2x,role=filename] ①
```

① L'attribut `role` peut être cumulé à celui de la taille



Figure 38. icônes formatées avec un rôle

Les rôles `filename` et `big-green` ne sont pas des [rôles intégrés](#), c'est-à-dire qu'ils ne sont pas prévus nativement dans le langage AsciiDoc. Il s'agit ici de rôles personnalisés ([chapitre 27, Mise en forme à l'aide de rôles personnalisés](#)).

Si vous souhaitez en savoir davantage sur les icônes, vous pouvez lire cette [page de la documentation](#).

# 13. Les références croisées

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 18:45 | Auteur : Emmanuel Ravrat

Durée de réalisation : 40min

## 13.1. Qu'est-ce qu'une référence croisée ?

Une **référence croisée** est tout simplement un lien interne. C'est-à-dire qu'en cliquant sur un lien depuis un document, vous vous rendez à un autre emplacement de ce document.

Les références croisées sont très utiles, notamment dans les longs documents. Cela améliore la navigation entre les parties.

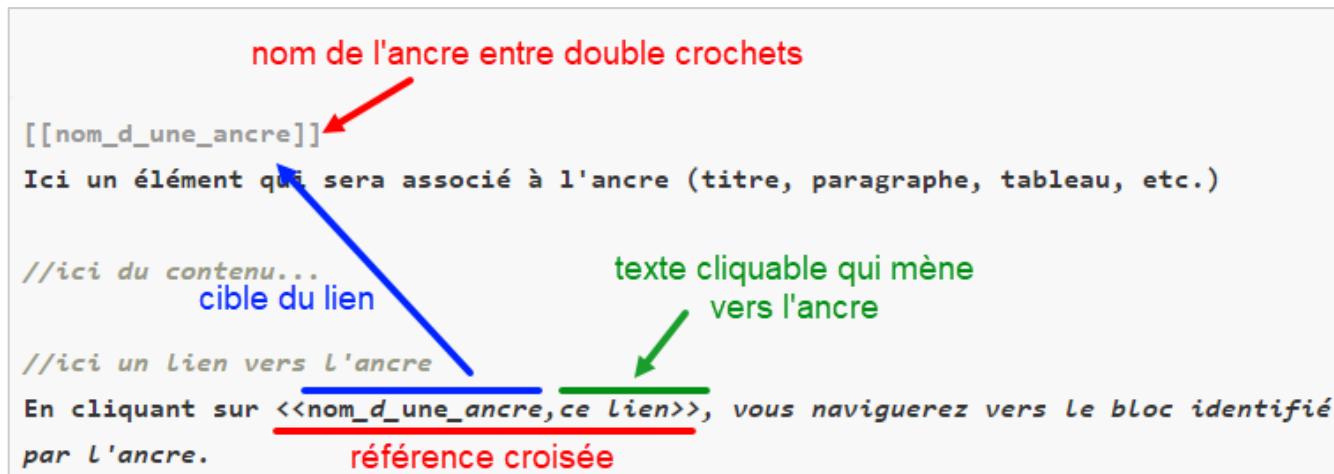
Une référence croisée est composée d'un lien et d'une ancre. Le lien pointe l'ancre.

Une **ancre** est un emplacement dans le document, identifié par **un nom unique**.

Voici le principe de fonctionnement :

- une ancre est placée sur un élément (un paragraphe, un titre, un mot, etc.). Une ancre est un **nom unique** encadrée de double crochets.
- un lien faire référence à l'ancre. Un lien est construit entre double chevrons << et >> et contient deux éléments : le nom de l'ancre à cibler et le texte cliquable séparé par une virgule.

Voici le schéma d'illustration :



Une ancre est toujours placée **immédiatement** avant l'élément à cibler. Quand l'élément est un bloc (un titre, un paragraphe, un tableau, une image, etc.), l'ancre est à placer sur la ligne qui précède le bloc.



Je conseille très fortement de ne pas utiliser d'espaces, de majuscules, de caractères accentués et de caractères spéciaux dans le nom de l'ancre.

Les espaces peuvent être remplacés par des underscores. Cela permet d'avoir une autocomplétion plus pertinente de la part de l'IDE.

C'est pénible à écrire mais grâce à votre IDE, vous allez pouvoir automatiser tout cela (voir le cours sur les lives templates)

Pour récapituler :

```
[[nom_d_une_ancre]]
```

Ici un élément qui sera associé à l'ancre (titre, paragraphe, tableau, etc.)

```
//ici du contenu...
```

```
//ici un lien vers l'ancre
```

En cliquant sur <<nom\_d\_une\_ancre,ce lien>>, vous naviguerez vers le bloc identifié par l'ancre.

En cliquant sur [ce lien](#), vous  
naviguerez vers le bloc identifié par  
l'ancre.

Figure 39. rendu d'une référence croisée



Une fois le nom d'une ancre défini, sa modification ne doit pas être faite à la légère. Le point de cours renommer une ancre doit être absolument respecté.

## 13.2. Référencer un titre

Voici un extrait d'un très long document sur la vie de Chuck Norris qui comme nous le savons est une référence pour superman.



L'objectif est de faire un lien situé tout à la fin du document qui permet de cibler le titre de la première partie de ce même document :

```
= Chuck Norris : l'homme, le mythe, la légende  
[[origine_legende_chuck]] ①  
== L'origine de la légende  
  
// [...]  
  
== Influence culturelle globale  
  
Vous pouvez relire la partie <>origine_legende_chuck<> ②  
  
//fin du document (8 500 000 lignes plus tard...)
```

① ancre associée au titre « Influence culturelle globale »

② lien qui pointe l'ancre donc le titre qui lui est associé.

**J'attire votre attention sur le fait que le lien n'a pas de texte cliquable de défini, c'est volontaire.**

**Vous pouvez relire la partie L'origine de la légende**

Figure 40. référence croisée avec texte du lien correspondant au titre ciblé

Vous pouvez remarquer que le texte du lien est automatiquement celui du titre ! C'est très pratique car si vous changez le titre (mais pas le nom de l'ancre), le texte du lien est automatiquement mis à jour !

En voici l'illustration en changeant le titre :

```
= Chuck Norris : l'homme, le mythe, la légende  
[[origine_legende_chuck]] ①  
== Là où tout a commencé ②  
  
// [...]  
  
== Influence culturelle globale  
  
Vous pouvez relire la partie <>origine_legende_chuck<> ②  
  
//fin du document (8 500 000 lignes plus tard...)
```

① Le nom de l'ancre n'a pas changé

② Le titre a été modifié

Le texte cliquable a été mis automatiquement à jour !

**Vous pouvez relire la partie [Là où tout a commencé](#)**

Figure 41. le texte du lien de la référence croisée a été mis à jour

 Si vous faites un lien vers une ancre qui n'existe pas dans le document principal depuis lequel le fichier pdf est généré, c'est le nom de l'ancre qui sera affichée à la place du texte du titre.

Il faut comprendre par "document principal" le document et tous les fichiers qu'il inclut (voir la partie  [title\_inclure\_des\_fichiers\_asciidoc]).

Le texte du lien peut être personnalisé par l'ajout d'un texte après le nom de l'ancre :

```
= Chuck Norris : l'homme, le mythe, la légende
```

```
[[origine_legende_chuck]]
```

```
== Là où tout a commencé
```

```
// [...]
```

```
== Influence culturelle globale
```

Vous pouvez relire la partie <>origine\_legende\_chuck, qui est tout au début du document>> ①

```
//fin du document (8 500 000 lignes plus tard...)
```

① Le texte du lien est précisé après la virgule qui suit le nom de l'ancre à cibler

**Vous pouvez relire la partie [qui est tout au début du document](#)**

Figure 42. référence croisée avec texte personnalisé

 Il est possible de référencer un titre sans avoir explicitement défini une ancre. Dans ce cas, le nom de l'ancre sera exactement le titre (si celui-ci contient au moins une majuscule ou un espace).

Je déconseille fortement cette pratique car un titre est souvent reformulé ce qui rend la référence vers cette ancre non fonctionnelle.

### 13.3. Référencer un bloc : paragraphe, tableau, liste, etc.

#### Ancrer un paragraphe

Référencer un paragraphe nécessite de poser une ancre sur la ligne qui précède immédiatement le paragraphe.

Même si le fait de laisser une ligne vide ne change rien entre l'ancre et le paragraphe, je recommande de ne pas le faire afin de mieux visualiser l'élément ancré.

```
[[anecdote_bibliotheque_chuck]] ①
Un jour, Chuck Norris est entré dans une bibliothèque et a demandé un livre sur
l'autodéfense. Le bibliothécaire, tremblant, lui a donné un miroir. Chuck Norris a
souri, et depuis, ce miroir est exposé dans un musée avec une plaque indiquant "L'arme
la plus puissante du monde".

// du contenu ici ...

//un lien vers l'anecdote de la bibliothèque
Vous pouvez lire le paragraphe qui reprend une <>anecdote_chuck, anecdote>> de Chuck
Norris ②
```

① ancre fixée au paragraphe

② référence croisée vers l'ancre avec en premier élément le nom de l'ancre et en second élément le texte cliquable.

You can read the paragraph which
contains a [cross-referenced anecdote](#) of Chuck Norris

Figure 43. référence croisée vers un paragraphe

#### Ancrer un tableau

```
// du contenu ici...

[[evaluation_competences_chuck]] ①
[%awidth,cols="2*a", options="header"] ②
|===
^.^| Compétence de Chuck
^.^| Niveau sur 10

| force pure
| 1547

| regard écrasant
| 3621
| ===
```

```
// du contenu ici...
```

Vous pouvez vous référer au <<evaluation\_competences\_chuck, tableau de compétences>> de Chuck. ③

① ancre fixée au tableau

② début d'un tableau AsciiDoc (voir le chapitre dédié pour apprendre à faire les tableaux)

③ référence croisée vers le tableau

**Vous pouvez vous référer au [tableau de compétences](#) de Chuck.**

Figure 44. référence croisée vers un tableau

### Ancrer une liste

```
// du contenu ici...
```

```
[[liste_competences_chuck]] ①
* force pure
* regard écrasant
```

```
// du contenu ici...
```

Vous pouvez consulter la <<liste\_competences\_chuck, liste de compétences>> de Chuck. ②

① ancre fixée à la liste

② référence croisée vers la liste

**Vous pouvez consulter la [liste de compétences](#) de Chuck.**

Figure 45. référence croisée vers une liste



Une fois que vous avez compris le principe, sachez que la technique est valable pour tous les éléments considérés comme des "blocs".

## 13.4. Référencer un élément en "ligne" : des mots, un item de liste, etc.

### Ancrer un item de liste ou une cellule de tableau

Pour ajouter une ancre à un item de liste, il faut placer l'ancre après le caractère marquant l'item et avant d'écrire le texte.

Voici un exemple avec un item de liste

```
// du contenu ici...  
  
[[liste_competences_chuck]] ①  
* force pure  
* [[competence REGARD]] regard écrasant ②  
  
// du contenu ici...  
Le regard est une des <<competence REGARD, compétences>> de Chuck. ③
```

① ancre fixée à la liste

② ancre placé sur un item de liste

③ référence croisée vers l'item de liste

Pour une cellule de tableau, c'est exactement le même principe. L'ancre doit être placée au début de la cellule :

```
// du contenu ici...  
  
[%awidth,cols="2*a", options="header"] ①  
|===  
^.^| Compétence de Chuck  
^.^| Niveau sur 10  
  
| [[force_chuck]] force pure ②  
| 1547  
  
| regard écrasant  
| 3621  
|===  
  
// du contenu ici...
```

Voir le niveau de compétence de <<force\_chuck, force pure>> dans le tableau des compétences. ③

① début d'un tableau AsciiDoc (voir le chapitre dédié pour apprendre à faire les tableaux)

② ancre fixée à la cellule

③ référence croisée vers la cellule

## Ancre dans une ligne de contenu

Dans tous les cas précédents, l'ancre a été déclarée au début de l'élément à cibler. Mais, il est tout à fait possible d'utiliser une ancre au milieu d'un texte.

Chuck Norris a souri, et depuis, [[miroir\_chuck]] ce miroir est exposé dans un musée avec une plaque indiquant "L'arme la plus puissante du monde". ①

```
// du contenu ici...
```

L'objet mystère de Chuck est son <<miroir\_chuck,miroir>> de Chuck. ②

① l'ancre est placée au sein du contenu

② le lien marqué par <> permet de naviguer vers l'ancre

Pour aller plus loin avec les références croisées, notamment les références croisées entre documents, vous pouvez lire [cette page de la documentation](#).

## 13.5. Voir où une ancre est utilisée et comment la renommer

### Voir où une ancre est utilisée

Les IDE JetBrains avec le plugin Asciidoc permettent de voir où une ancre est utilisée. Cela n'est vrai que si l'ancre à un nom unique.

Pour voir où une ancre est utilisée, il faut maintenir la touche **CTRL** enfoncée et cliquer sur le nom de l'ancre.

Si votre ancre n'est utilisée que par un seul lien, vous naviguerez automatiquement vers l'emplacement, même si c'est dans un autre fichier (dès lors que le projet ouvert dans votre IDE contient ce fichier).

Si votre ancre est utilisée à différents endroits, même dans différents fichiers, vous aurez une liste de ces utilisations et vous pourrez cliquer sur celle qui vous intéresse le cas échéant.

Créez des liens qui pointent la même ancre et faites le test.

### Renommer une ancre

Il faut bien comprendre que si vous changez le nom d'une ancre, tous les liens qui utilisent le nom de cette ancre ne fonctionneront plus.

Si vous changez le nom de l'ancre, il faudra alors modifier tous les liens qui utilise ce nom ! Ce peut être un travail fastidieux et source d'erreurs et d'oublis.

Heureusement, si vous utilisez un IDE JetBrains et le plugin AsciiDoc, il est possible de renommer une ancre et que cela soit répercuté sur tous les liens qui pointent cette ancre.

Pour renommer une ancre, faites un clic droit sur le nom de l'ancre puis faire **Refactor > Rename**, modifiez le nom et appliquer les modifications. Votre IDE va mettre à jour les fichiers qui référencent cette ancre **s'ils figurent dans le projet ouvert dans votre IDE**. C'est-à-dire qu'il va modifier lui-même l'ancien nom pour le nouveau nom.

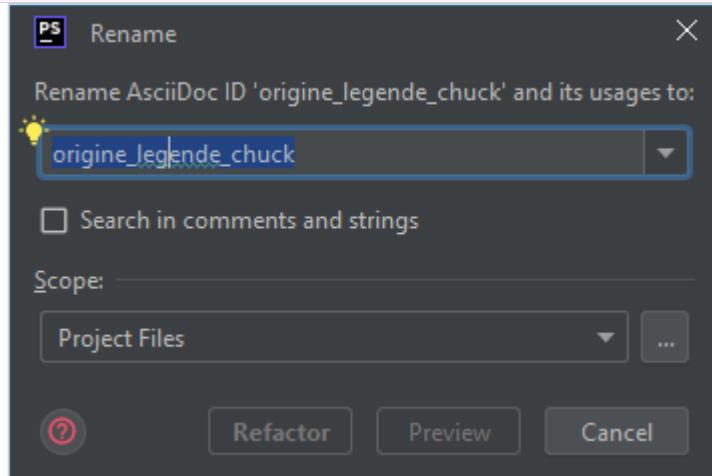


Figure 46. renommer une ancre avec un IDE Jetbrains

Créez des liens qui pointent la même ancre et renommez cette ancre en utilisant la refactorisation.

# 14. Les blocs d'avertissement (admonitions)

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 19:00 | Auteur : Emmanuel Ravrat

Durée de réalisation : 25min

## 14.1. Qu'est-ce qu'un bloc d'avertissement ?

Un **bloc d'avertissement** est un contenu assez court sur lequel une attention particulière doit être portée.



Un bloc d'avertissement doit porter sur un contenu atomique. Cela signifie qu'une seule idée, un seul concept, une seule remarque, doit figurer dans le bloc.

S'il y en a plusieurs, alors écrivez plusieurs blocs d'avertissement.

Il y a 5 types d'avertissement avec chacun une étiquette désignant un niveau d'attention à porter au contenu :

- **NOTE** : est utilisée pour ajouter des informations supplémentaires ou des commentaires qui ont un intérêt particulier pour le lecteur.
- **TIP** : utilisée pour fournir des conseils ou des astuces pratiques qui peuvent aider le lecteur à mieux comprendre ou à mieux utiliser l'information.
- **CAUTION** : utilisée pour avertir le lecteur d'une situation qui pourrait entraîner des problèmes ou des erreurs si elle n'est pas gérée correctement. C'est un avertissement modéré.
- **WARNING** : utilisée pour signaler un danger potentiel ou une situation qui pourrait avoir des conséquences gênantes si elle est ignorée.
- **IMPORTANT** : utilisée pour souligner des informations cruciales que le lecteur doit absolument connaître.

## 14.2. Ecrire des blocs d'avertissement

Un bloc d'avertissement s'écrit de la façon suivante :

[étiquette] ①

==== ②

Ici le message.

N'importe quel contenu peut être écrit dans un bloc d'avertissement (des paragraphes, des images, des listes, des tableaux, des blocs de code, etc.)

==== ③

① remplacer l'étiquette par **NOTE**, **TIP**, **CAUTION**, **WARNING**, **IMPORTANT**

② début du bloc d'avertissement marqué par 4 caractères =

③ fin du bloc d'avertissement marqué par 4 caractères =

Voici le rendu des 5 blocs d'avertissement (vous n'aurez pas tout de suite les icônes) :

- bloc d'avertissement de type NOTE

[NOTE]

====

Ceci est avertissement de type 'NOTE'.

Il est utilisée pour ajouter des informations supplémentaires ou des commentaires qui ont un intérêt particulier pour le lecteur.

====

Ceci est avertissement de type NOTE.

**NOTE**

Il est utilisée pour ajouter des informations supplémentaires ou des commentaires qui ont un intérêt particulier pour le lecteur.

Figure 47. rendu d'un avertissement de type NOTE

- bloc d'avertissement de type TIP

[TIP]

====

Ceci est avertissement de type 'TIP'.

Ce bloc contient un message contenant une astuce ou un conseil.

====

Ceci est avertissement de type TIP.

**TIP**

Ce bloc contient un message contenant une astuce ou un conseil.

Figure 48. rendu d'un avertissement de type NOTE

- bloc d'avertissement de type CAUTION

[CAUTION]

====

Ceci est avertissement de type 'CAUTION'.

Ce bloc est utilisé pour avertir le lecteur d'une situation qui pourrait entraîner des problèmes ou des erreurs si elle n'est pas gérée correctement.

====

<b>CAUTION</b>	Ceci est avertissement de type <b>CAUTION</b> .  Ce bloc est utilisé pour avertir le lecteur d'une situation qui pourrait entraîner des problèmes ou des erreurs si elle n'est pas gérée correctement.
----------------	--

Figure 49. rendu d'un avertissement de type CAUTION

- bloc d'avertissement de type **WARNING**

<b>[WARNING]</b> =====	Ceci est avertissement de type 'WARNING'.  Ce bloc est utilisé pour signaler un danger potentiel ou une situation qui pourrait avoir des conséquences gênantes si elle est ignorée. =====
---------------------------	--

<b>WARNING</b>	Ceci est avertissement de type <b>WARNING</b> .  Ce bloc est utilisé pour signaler un danger potentiel ou une situation qui pourrait avoir des conséquences gênantes si elle est ignorée.
----------------	---

Figure 50. rendu d'un avertissement de type WARNING

- bloc d'avertissement de type **IMPORTANT**

<b>[IMPORTANT]</b> =====	Ceci est avertissement de type 'IMPORTANT'.  Ce bloc est utilisé pour souligner des informations cruciales que le lecteur doit absolument connaître. =====
-----------------------------	---

<b>IMPORTANT</b>	Ceci est avertissement de type <b>IMPORTANT</b> .  Ce bloc est utilisé pour souligner des informations cruciales que le lecteur doit absolument connaître.
------------------	--

Figure 51. rendu d'un avertissement de type IMPORTANT

Les blocs d'avertissement peuvent être écrits en ligne :



TIP: Ceci est avertissement de type 'TIP' déclaré sur une ligne.

**TIP**

Ceci est avertissement de type **TIP**.

Figure 52. rendu d'un avertissement déclaré sur une ligne

Je déconseille son usage car ce n'est utile que pour les messages ne faisant qu'une seule ligne. De plus, c'est moins visible et lisible dans le fichier AsciiDoc que la syntaxe avec les signes **====** qui délimitent clairement le début et la fin du bloc.

Vous devez vous dire que c'est sympa les blocs d'avertissement mais que ce serait plus « cool » d'avoir des **icônes d'avertissement** à la place des mots **NOTE**, **TIP**, **CAUTION**, **WARNING**, **IMPORTANT**.

Cela va être réglé en un tour de main avec l'**attribut de document :icons:** associée à la valeur **font**.

Puisque **:icons:** est un attribut de document, il doit être placé immédiatement sous le titre principal du document ou entre ou sous les autres attributs de document.

Voici un exemple :

```
= Chuck Norris : l'homme, le mythe, la légende ①
:toc: ②
:toclevels: 5 ②
:icons: font ③
```

① titre principal du document

② d'autres attributs de document

③ attribut de document indiquant que les icônes de la police utilisée doivent illustrer les blocs d'avertissement.

Une fois cet attribut déclaré avec la valeur **font**, vous devriez voir le même rendu que ceux-ci :

Ceci est avertissement de type **NOTE**.



Il est utilisée pour ajouter des informations supplémentaires ou des commentaires qui ont un intérêt particulier pour le lecteur.

Figure 53. rendu d'un bloc d'avertissement de type **NOTE**

Ceci est avertissement de type **TIP**.



Ce bloc contient un message contenant une astuce ou un conseil.

Figure 54. rendu d'un bloc d'avertissement de type **TIP**



Ceci est avertissement de type **CAUTION**.

Ce bloc est utilisé pour avertir le lecteur d'une situation qui pourrait entraîner des problèmes ou des erreurs si elle n'est pas gérée correctement.

Figure 55. rendu d'un bloc d'avertissement de type **CAUTION**



Ceci est avertissement de type **WARNING**.

Ce bloc est utilisé pour signaler un danger potentiel ou une situation qui pourrait avoir des conséquences gênantes si elle est ignorée.

Figure 56. rendu d'un bloc d'avertissement de type **WARNING**



Ceci est avertissement de type **IMPORTANT**.

Ce bloc est utilisé pour souligner des informations cruciales que le lecteur doit absolument connaître.

Figure 57. rendu d'un bloc d'avertissement de type **IMPORTANT**

Si vous voulez en savoir plus sur les blocs (car il n'existe pas que les blocs d'avertissement), vous pouvez lire la [page de la documentation dédiée aux blocs](#).

# 15. Les blocs de code source

Version 1 | Dernière mise à jour : 26/08/2024 à 19:32 | Auteur : Emmanuel Ravrat

Durée de réalisation : 30min

## 15.1. Le bloc de liste

Le **bloc de liste AsciiDoc** est un bloc dont le contenu est rendu avec une police à espacement fixe et tel qu'il est saisi. C'est-à-dire que les retours à la ligne sont conservés tout comme les caractères qui servent habituellement à faire la mise en forme.

*Ne me demandez pas pourquoi cela s'appelle un bloc de liste, je n'ai pas trouvé la réponse.*

Un bloc de liste commence par 4 tirets et se termine par 4 tirets :

```
---- ①
ici du contenu dans un bloc de liste.
Si je retourne
à la ligne et que j'utilise des caractères
pour mettre *en gras* ou en _italique_, le contenu
est restitué tel qu'il a été saisi.
---- ②
```

① début du bloc de liste

② fin du bloc de liste

Le rendu est identique au code contenu dans le bloc de liste :

```
ici du contenu dans un bloc de liste.
Si je retourne
à la ligne et que j'utilise des caractères
pour mettre *en gras* ou en _italique_, le contenu
est restitué tel qu'il a été saisi.
```

J'utilise le bloc de liste pour mettre en avant des messages d'erreur, montrer du texte tel que j'ai pu le copier dans le presse-papier, etc.

Ce type de bloc prend tout son intérêt lorsqu'il est spécialisé (voir le point suivant).

## 15.2. Du code dans un bloc de liste

Un **bloc de liste** peut être utilisé pour y écrire du code source.

Voici un bloc de liste reconnaissable par ses 4 tirets qui marque le début et la fin du bloc :

```
----
```

Du contenu écrit dans un bloc de liste dont le début du bloc est marqué par quatre tirets tout comme sa fin.

----

Du **code source** désigne des lignes écrites dans un langage de programmation.

Le bloc de liste peut contenir du code source concernant un langage de programmation en particulier. Dans ce cas, il faut indiquer le "style" du bloc qui est **source** suivi du nom du langage contenu dans le bloc.

Voici un exemple de bloc de code source contenant du HTML :

```
[source,html] ①
-----
<p>Ceci est un paragraphe avec un mot <strong>important</strong></p>
-----
```

① l'attribut **source** indique que le bloc de liste contient du code source et **html** précise le langage contenu dans le bloc.

```
<p>Ceci est un paragraphe avec un mot <strong>important</strong></p>
```

Figure 58. rendu d'un bloc de liste spécialisé comme source de code

AsciiDoc prend en charge de nombreux langages, tel que javascript, php, java, etc.

Voici un autre exemple de code source écrit en Java :

```
[source,java]
-----
System.out.println("Hello, World!");
-----
```

```
System.out.println("Hello, World!");
```

Figure 59. rendu d'un bloc de code contenant du java



La valeur de la langue source à spécifier après l'attribut **source** est généralement le nom du langage écrit en minuscules (javascript, java, php, html, css, etc.).

## 15.3. Autres fonctionnalités liées aux blocs de code

### 15.3.1. Afficher les numéros de lignes



L'affichage des numéros de ligne est ajouté par le surlieur de syntaxe installé sur votre système. Si vous n'avez pas les numéros, alors jetez un oeil à cette page de la documentation qui montre comment utiliser le surlieur nommé Rouge.

Pour ajouter les numéros de lignes du bloc de code, ajoutez l'option `linenums` juste après l'attribut `source` en les séparant par un %:

```
[source%linenums,php] ①
-----
$age = 22;

if(18>$age){
echo 'Vous êtes mineur';
}
-----
```

① L'ajout de l'option `linenums` permet de rendre les numéros de ligne

```
1 $age = 22;
2
3 if(18>$age){
4 echo 'Vous êtes mineur';
5 }
```

Figure 60. affichage des numéros de ligne

### 15.3.2. Mettre en surbrillance une ou plusieurs lignes

Pour surligner des lignes, il faut avoir activé l'affichage des numéros de ligne avec l'option `linenums`. Une fois cela fait, il faut spécifier le ou les numéros de ligne à surligner avec l'attribut `highlight` :

Code AsciiDoc	Rendu
<h3>Surligner une seule ligne</h3> <pre>[source%linenums,php,highlight=3] ① ----- \$age = 22;  if(18&gt;\$age){     echo 'Vous êtes mineur'; } -----</pre> <p>① La ligne 3 est surlignée</p>	<p>La ligne 3 est surlignée :</p> <pre>1 \$age = 22; 2 3 if(18&gt;\$age){ 4     echo 'Vous êtes mineur'; 5 }</pre>
<h3>Surligner plusieurs lignes</h3> <p>Pour surligner plusieurs lignes, il faut les lister en les séparant par un point-virgule.</p> <pre>[source%linenums,php,highlight=3;5] ① ----- \$age = 22;  if(18&gt;\$age){     echo 'Vous êtes mineur'; } -----</pre> <p>① Les lignes 3 et 5 sont surlignées</p>	<p>Les lignes 3 et 5 sont surlignées :</p> <pre>1 \$age = 22; 2 3 if(18&gt;\$age){ 4     echo 'Vous êtes mineur'; 5 }</pre>
<h3>Surligner une plage de lignes</h3> <p>Pour surligner plusieurs lignes qui se suivent, il faut indiquer la première ligne et la dernière ligne de la plage en les séparant par un double point.</p> <pre>1 [source%linenums,php,highlight=3..5] ① 2 ----- 3 \$age = 22; 4 5 if(18&gt;\$age){ 6     echo 'Vous êtes mineur'; 7 } 8 -----</pre> <p>① Les lignes 3 à 5 sont surlignées</p>	<p>Les lignes 3 à 5 sont surlignées :</p> <pre>1 \$age = 22; 2 3 if(18&gt;\$age){ 4     echo 'Vous êtes mineur'; 5 }</pre>

Code AsciiDoc	Rendu
<p><b>Combiner les approches</b></p> <p>Il est possible de surligner une ligne isolées et une suite de lignes.</p> <div data-bbox="165 428 779 729"><pre>[source%linenums,php,highlight=1;3..5] ① ----- \$age = 22;  if(18&gt;\$age){     echo 'Vous êtes mineur'; } -----</pre></div> <p>① La ligne 1 et les lignes 3 à 5 sont surlignées</p>	<p>La ligne 1 et les lignes 3 à 5 sont surlignées :</p> <div data-bbox="1060 458 1378 691"><pre>1 \$age = 22; 2 3 if(18&gt;\$age){ 4     echo 'Vous êtes mineur'; 5 }</pre></div>

### 15.3.3. Contrôler l'indentation du code dans le bloc

Le code source n'est pas toujours écrit contre la marge de gauche. Cela arrive notamment lorsque l'on copie colle du code issu d'un fichier. Cela arrive également lorsqu'on inclu du [code issu d'un fichier externe](#).

C'est ce que l'on appelle l'**indentation du code**. Il y a un "espace" entre le bord gauche constitué de une à plusieurs tabulations.

Dans l'exemple ci-après, le code n'est pas placé contre le bord gauche car il est indenté (décalé du bord).

```
[source,php]
-----
$age = 22;

if(18>$age{
    echo 'Vous êtes mineur';
}
-----
```

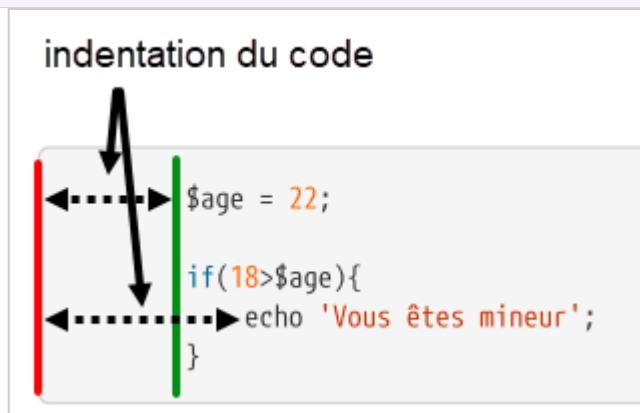


Figure 61. rendu du code source indenté dans le bloc de code

Cela peut être corrigé grâce à l'attribut `indent` suivi d'une valeur représentant le nombre d'indentations à laisser.

```
[source,php,indent=0] ①
```

```
-----
$age = 22;

if(18>$age){
    echo 'Vous êtes mineur';
}
-----
```

① `indent = 0` signifie que le code doit être situé à 0 tabulation du bord gauche.

```
$age = 22;

if(18>$age){
    echo 'Vous êtes mineur';
}
```

Figure 62. correction de l'indentation du code avec l'attribut `indent`



Pour augmenter la valeur de l'indentation, il suffit de mettre une valeur d'attribut `indent` supérieur à 0. Plus la valeur sera grande et plus importante sera l'indentation.

### 15.3.4. Les numéros de légende (callouts)

Les **callouts** ou **numéros de légende** sont des numéros qui permettent de faire un renvoi sous le bloc de code. C'est très utile pour donner des explications supplémentaires.

Un numéro s'ajoute **en fin de ligne** et est placé entre chevrons : `<numero>` dans un commentaire :

```
[source,php]
```

```
-----
$age = 22; // <1>
```

```
if(18>$age){ // <2>
    echo 'Vous êtes mineur'; // <3>
}
-----
<1> affectation d'une variable
<2> évaluation de la condition
<3> code exécuté lorsque la condition est vraie
```



Placer le callout dans un commentaire n'est pas obligatoire mais je vous conseille très vivement de le faire. Ainsi, si vous faites un copier-coller du code pour l'exécuter ou le compiler, les callouts seront tout simplement ignorés.

```
$age = 22; ①

if(18>$age){ ②
    echo 'Vous êtes mineur'; ③
}
```

- ① affectation de la variable
- ② évaluation de la condition
- ③ code exécuté lorsque la condition est vraie.

Figure 63. utilisation de callouts ou numéro de légende

Il faut adapter le commentaire au langage spécifié dans le bloc de code.

Voici un exemple avec du HTML :



```
[source,html]
-----
<p>Ceci est un paragraphe</p> <!--1-->
-----
<1> la balise 'p' contient un paragraphe
```

```
<p>Ceci est un paragraphe</p> ①
```

- ① la balise p contient un paragraphe

Figure 64. rendu d'un callout dans un bloc de code HTML



Les numéros de légende doivent débuter à 1 puis être incrémentés de 1 en 1.

Si vous faites face à des besoins particuliers en matière de « callouts », vous pouvez vous référer à

la [page de documentation dédiée](#).

Si vous voulez en savoir plus sur les blocs (car il n'existe pas que les blocs de code), vous pouvez lire la [page de la documentation dédiée aux blocs](#).

# 16. Inclure des fichiers de code

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 19:40 | Auteur : Emmanuel Ravrat

Durée de réalisation : 45min

## 16.1. Le bloc de code : rappel

Un bloc de code est un bloc littéral spécialisé.

Un bloc littéral est un bloc dont le contenu est rendu à l'identique de ce qu'il est dans le fichier AsciiDoc.

Un bloc littéral est encadré par 4 tirets. L'ajout d'un attribut `source` permet d'indiquer qu'il s'agit d'un bloc de code. Le langage est alors précisé après cet attribut :

Voici un bloc de code Java :

```
[source,java] ①
---- ②
public class ChuckNorris {

    private String prenom = "Chuck";
    private String nom = "Norris";

    public String compterInfini() {
        return prenom + " " + nom + " a compté jusqu'à l'infini. Deux fois.";
    }

    public String eteindreLeNoir() {
        return "Quand " + prenom + " " + nom + " entre dans une pièce, il n'allume pas
la lumière. Il éteint le noir.";
    }

    public String diviserParZero() {
        return prenom + " " + nom + " peut diviser par zéro.";
    }

    @Override
    public String toString() {
        return compterInfini() + "\n" + eteindreLeNoir() + "\n" + diviserParZero();
    }

    public static void main(String[] args) {
        ChuckNorris chuck = new ChuckNorris();
        System.out.println(chuck);
    }
}
---- ③
```

① le bloc littéral est un bloc de code source contenant du code java

② début du bloc littéral

③ fin du bloc littéral

Rendu du bloc de code :

```
public class ChuckNorris {

    private String prenom = "Chuck";
    private String nom = "Norris";

    public String compterInfini() {
        return prenom + " " + nom + " a compté jusqu'à l'infini. Deux fois.";
    }

    public String eteindreLeNoir() {
        return "Quand " + prenom + " " + nom + " entre dans une pièce, il n'allume pas la lumière. Il éteint le noir.";
    }

    public String diviserParZero() {
        return prenom + " " + nom + " peut diviser par zéro.";
    }

    @Override
    public String toString() {
        return compterInfini() + "\n" + eteindreLeNoir() + "\n" + diviserParZero();
    }

    public static void main(String[] args) {
        ChuckNorris chuck = new ChuckNorris();
        System.out.println(chuck);
    }
}
```

Cette fonctionnalité est bien pratique pour écrire du code dans une documentation. Cependant, le code provient généralement des fichiers d'un véritable programme. A chaque fois que le programme évolue, il faut de nouveau copier-coller le contenu dans le fichier AsciiDoc.

Cela peut être évité en faisant une inclusion du fichier de code.

## 16.2. Inclure un fichier de code

L'inclusion d'un fichier de code se fait avec la macro `include::` suivi du chemin relatif (ou absolu) du fichier à inclure.

Créez un dossier `code` dans le répertoire parent du fichier AsciiDoc que vous êtes en train d'éditer. Puis dans ce dossier, créez un fichier `chuck.java` qui contient le code source qui figure dans le bloc

de code précédent.

Une fois le fichier à inclure en place, il faut spécifier le chemin relatif vers celui-ci :

```
[source,java]
-----
\include::code/chuck.java[] ①
-----
```

- ① Le bloc de code source va contenir l'intégralité du code source du fichier `chuck.java`. Il ne faut pas oublier les crochets en fin de macro.

Le **contenu du fichier est inclus dans son intégralité**. Toute modification dans le fichier source est répercutée dans la documentation ! C'est très pratique et cela permet de tester son code puisque le fichier est un fichier de code comme les autres.

Rendu de la classe Java `ChuckNorris` après inclusion du fichier `code/chuck.java` :

```
public class ChuckNorris {

    private String prenom = "Chuck";

    private String nom = "Norris";

    public String compterInfini() {
        return prenom + " " + nom + " a compté jusqu'à l'infini. Deux fois.";
    }

    public String eteindreLeNoir() {
        return "Quand " + prenom + " " + nom + " entre dans une pièce, il n'allume pas la lumière. Il éteint le noir.";
    }

    public String diviserParZero() {
        return prenom + " " + nom + " peut diviser par zéro.";
    }

    @Override
    public String toString() {
        return compterInfini() + "\n" + eteindreLeNoir() + "\n" + diviserParZero();
    }

    public static void main(String[] args) {
        ChuckNorris chuck = new ChuckNorris();
        System.out.println(chuck);
    }
}
```

## 16.3. Inclure une partie d'un fichier de code

### 16.3.1. Inclure une seule région de code

Il arrive fréquemment qu'une documentation ne porte que sur une portion du code d'un fichier.

AsciiDoc permet de créer des "zones de code" délimitées par un nom. Ces zones sont des régions taguées.

Une **région de code taguée** est constituée d'un commentaire de début et d'un commentaire de fin qui permettent respectivement d'identifier le début et la fin d'un ensemble de lignes de code dans un fichier. Cela constitue une "région". Chaque région à un nom qui permet de l'identifier et d'accéder à son contenu.

Une région commence par une ligne de commentaire commençant par la macro `tag::` suivi du nom de la région suivie de crochets. La fin d'une région est marquée par une ligne de commentaire commençant par la macro `end::` suivi du nom de la région à fermer suivie de crochets.

Voici le fichier précédent, cette fois avec 6 régions définies :

```
// tag::class_ChuckNorris[]
public class ChuckNorris {

    private String prenom = "Chuck";

    //tag::property_nom[]
    private String nom = "Norris";
    //end::property_nom[]

    //tag::method_compterInfini[]
    public String compterInfini() {
        return prenom + " " + nom + " a compté jusqu'à l'infini. Deux fois.";
    }
    //end::method_compterInfini[]

    //tag::method_eteindreLeNoir[]
    public String eteindreLeNoir() {
        return "Quand " + prenom + " " + nom + " entre dans une pièce, il n'allume pas la lumière. Il éteint le noir.";
    }
    //end::method_eteindreLeNoir[]

    //tag::method_diviserParZero[]
    public String diviserParZero() {
        return prenom + " " + nom + " peut diviser par zéro.";
    }
    //end::method_diviserParZero[]

    // tag::methodes_pour_tests[]
```

```
@Override  
public String toString() {  
    return compterInfini() + "\n" + eteindreLeNoir() + "\n" + diviserParZero();  
}  
  
public static void main(String[] args) {  
    ChuckNorris chuck = new ChuckNorris();  
    System.out.println(chuck);  
}  
// end::methodes_pour_tests[]  
}  
// end::class_ChuckNorris[]
```



Les régions peuvent s'imbriquer mais ne peuvent pas se chevaucher.



Je conseille de nommer les régions de façon à comprendre ce qu'elles contiennent.

Nous avons donc 6 régions. Nous allons demander à n'avoir que la région nommée `class_ChuckNorris`. L'attribut `tags` (entre les crochets de la macro `include::`) permet de lister la ou les régions que l'on souhaite rendre :

```
[source,java]  
----  
include::code/chuck.java[tags=class_ChuckNorris] ①  
----
```

① Le contenu de la région nommée `class_ChuckNorris` sera rendu dans le document final

*rendu de la région class\_ChuckNorris*

```
public class ChuckNorris {  
  
    private String prenom = "Chuck";  
  
    private String nom = "Norris";  
  
    public String compterInfini() {  
        return prenom + " " + nom + " a compté jusqu'à l'infini. Deux fois.";  
    }  
  
    public String eteindreLeNoir() {  
        return "Quand " + prenom + " " + nom + " entre dans une pièce, il n'allume pas  
la lumière. Il éteint le noir.";  
    }  
  
    public String diviserParZero() {  
        return prenom + " " + nom + " peut diviser par zéro.";
```

```
}

@Override
public String toString() {
    return compterInfini() + "\n" + eteindreLeNoir() + "\n" + diviserParZero();
}

public static void main(String[] args) {
    ChuckNorris chuck = new ChuckNorris();
    System.out.println(chuck);
}
}
```

Comme vous pouvez le voir, les régions "enfants" de la région rendue sont également affichées. Cela est normal puisqu'elles se trouvent à l'intérieur de la région à rendre.

Faisons un autre test en incluant seulement la région `method_compterInfini` :

```
[source,java]
-----
include::code/chuck.java[tags=method_compterInfini] ①
-----
```

① Seul le contenu de la région nommée `method_compterInfini` sera rendu dans le document final

*rendu de la région method\_compterInfini*

```
public String compterInfini() {
    return prenom + " " + nom + " a compté jusqu'à l'infini. Deux fois.";
}
```

Cependant, il y a un petit défaut. Le code est espacé d'une tabulation dans le fichier source et cela est rendu dans le document.

Pour annuler cet espace (appelé "indentation"), nous pouvons ajouter l'attribut `indent` avec la valeur `0` dans la déclaration du bloc de code :

```
[source,java,indent=0] ①
-----
public String compterInfini() {
    return prenom + " " + nom + " a compté jusqu'à l'infini. Deux fois.";
}
-----
```

① l'attribut `indent` permet de préciser la valeur d'indentation du code rendu. La valeur `0` précise qu'il ne faut pas d'indentation.

Si vous n'avez pas compris cette histoire d'indentation, vous pouvez relire le point [partie 15.3.3](#),

## “Contrôler l’indentation du code dans le bloc”.

*suppression de l’indentation qui existe dans le fichier source*

```
public String compterInfini() {  
    return prenom + " " + nom + " a compté jusqu'à l'infini. Deux fois.";  
}
```

### 16.3.2. Inclure plusieurs régions de code

Vous pouvez spécifier plusieurs régions en les séparant par un point virgule :

```
[source,java, indent=0]  
----  
include::code/chuck.java[tags=method_compterInfini;method_diviserParZero] ①  
----
```

① Seules les régions `method_compterInfini` et `method_diviserParZero` seront rendues dans le document

*rendu des seules régions `method_compterInfini` et `method_diviserParZero`*

```
public String compterInfini() {  
    return prenom + " " + nom + " a compté jusqu'à l'infini. Deux fois.";  
}  
public String diviserParZero() {  
    return prenom + " " + nom + " peut diviser par zéro.";  
}
```



Plusieurs régions peuvent avoir le même nom. Lorsque ce nom est inclus, les régions ayant ce nom sont rendues.

### 16.3.3. Exclure des régions enfant

Lorsque nous avons inclus la région `class_ChuckNorris`, nous avons récupéré tout le contenu de celle-ci. C'est le comportement normal et attendu. Mais si cette région contient des régions enfants que nous ne souhaitons pas rendre, il faut pouvoir les exclure.

Voici comment inclure la région `class_ChuckNorris` sans les régions enfant :

```
[source,java, indent=0]  
----  
include::code/chuck.java[tags=class_ChuckNorris;!*] ①  
----
```

① La région `class_ChuckNorris` est rendue puis toutes les autres régions sont exclues grâce à `!*`. Le caractère `*` indique "inclure toutes les régions" et le caractère `!` annule le caractère qui suit. Cela

revient à exclure tout ce qui n'a pas déjà été inclus.

seule la région `class_ChuckNorris` est rendue

```
public class ChuckNorris {  
    private String prenom = "Chuck";  
}
```



Toutes les lignes vides contenues dans la région filtrée sont affichées ce qui explique le "vide" entre la ligne de code et l'accolade fermante }

Dans le cas que nous venons de voir, toutes les régions enfants ont été exclues. Si nous voulons ajouter une ou plusieurs régions enfants, il suffit de les spécifier après l'exclusion :

```
[source, indent=0]  
----  
include::code/chuck.java[tags=class_ChuckNorris;!*;method_compterInfini] ①  
----
```

① la région `method_compterInfini` sera rendue car elle est explicitement demandée après avoir été exclue.

ajout d'une région exclue

```
public class ChuckNorris {  
  
    private String prenom = "Chuck";  
  
    public String compterInfini() {  
        return prenom + " " + nom + " a compté jusqu'à l'infini. Deux fois."; }  
}
```



En jouant avec les inclusions et les exclusions (et avec une certaine habitude), il est possible d'affiner le rendu en fonction de ses besoins.

Si vous voulez en apprendre davantage sur le filtrage des régions, vous pouvez lire la [page de documentation dédiée à l'inclusion de régions balisées](#). Effectivement, il est possible d'aller bien plus loin dans le filtrage des régions.

Si jamais vous aviez besoin d'inclure du code issu d'un dépôt distant (par exemple depuis un dépôt github ou gitlab), sachez que c'est possible. Voici la [page de documentation](#) de cette fonctionnalité.

Enfin, si vous avez besoin d'inclure le contenu d'un fichier texte, vous pouvez le faire en spécifiant des numéros de lignes ou des plages de lignes comme indiqué dans [cette page de la documentation](#).

# 17. Les tableaux

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 20:55 | Auteur : Emmanuel Ravrat

Durée de réalisation : 2h 15min

## 17.1. Comment travailler ce chapitre ?

Ce chapitre est l'un des plus moins simples concernant AsciiDoc.

Toutefois, vous allez apprendre à créer des tableaux avec ce langage.

Créer un tableau simple ne pose pas de difficulté. Mais si vous souhaitez utiliser des fonctionnalités d'entête, d'alignement, de mise en forme, etc., cela se complexifie un peu.

Je pense qu'il faut lire ce chapitre toujours en testant le rendu dans son IDE et faire des tests en modifiant des valeurs. Comme cela, vous savez (dans un coin de votre tête) que telle ou telle chose est possible. Le jour où vous aurez un besoin spécifique, vous pourrez revenir dans ce chapitre et piocher dans les parties qui vous seront utiles.

Encore une fois, AsciiDoc offre de quoi satisfaire de nombreux besoins. Vous ne chercherez plus à éviter à faire des tableaux !

## 17.2. Déclarer un tableau et ses dimensions

### 17.2.1. Déclarer le début et la fin d'un tableau

En AsciiDoc, un tableau est délimité par `|==`.

|== ①

|== ②

① début du tableau

② fin du tableau

### 17.2.2. Déterminer le nombre de colonnes

Il faut ensuite déterminer le nombre de colonnes avec l'attribut de bloc `cols` :

[cols="1,1,1"] ①

|==

|==

① le tableau sera composé de 3 colonnes car trois valeurs sont définies dans l'attribut `cols`

Les 3 nombres séparés par une virgule `1,1,1` sont des spécificateurs de colonne.

Le nombre de colonnes peut être spécifié à l'aide du caractère **\*** qui est un multiplicateur :

```
[cols="3*"] ①  
|===  
|==
```

① tableau de 3 colonnes

### 17.2.3. Ajouter des lignes

Une **ligne d'un tableau** est constitué de « cellules ».

Une **cellule de tableau** est marquée par le caractère pipe | (ALTGR + touche alphanumérique 6)

Ajout d'une ligne à notre tableau :

```
[cols="1,1,1"]  
|===  
|A |B |C ①  
|==
```

① tableau avec une seule ligne de trois cellules

A	B	C
---	---	---

Ajout d'une seconde ligne à notre tableau :

```
[cols="1,1,1"]  
|===  
|A |B |C  
|D |E |F ①  
|==
```

① seconde ligne ajoutée

A	B	C
D	E	F

Pour un petit tableau avec peu de données dans les cellules, cette façon d'écrire est acceptable. Mais je reviens sur un principe de base avec AsciiDoc : **on écrit qu'une seule phrase par ligne**. Cela signifie qu'il ne faut écrire qu'une seule cellule par ligne !

Voici le même tableau compte tenu de ce principe :

```
[cols="1,1,1"]  
|===  
|A
```

```
|B  
|C  
|D  
|E  
|F  
|==
```

Certes, le tableau est moins lisible dans le fichier AsciiDoc, mais sa mise à jour sera bien plus aisée (croyez-en mon expérience !).

Vous pouvez utiliser les commentaires et les lignes vides pour faciliter la lecture de votre tableau :

```
[cols="1,1,1"]
```

```
|==  
|A  
|B  
|C
```

```
// -----①
```

```
|D  
|E  
|F  
|==
```

① commentaire visuel permettant de bien marquer la séparation entre deux lignes. J'ai ajouté une ligne vide avant et après pour que cela soit encore plus visible mais vous faites selon votre envie.

Lorsque j'ai des tableaux avec pas mal de lignes, je spécifie le nombre de colonnes dans le commentaire de séparation :

```
[cols="1,1,1"]
```

```
|==  
|A  
|B  
|C
```

```
// -----td3-----①
```

```
|D  
|E  
|F  
|==
```

① le commentaire me permet de voir qu'il y a une ligne de tableau de trois colonnes (« td » pour « table data » ce qui correspond aux données de la cellule). Je n'ai pas besoin de voir la valeur affectée à l'attribut `cols`. Ceci peut paraître fastidieux mais

peut être automatisé grâce à des lives template (voir le cours sur les lives template)



Une cellule peut contenir n'importe quel élément : un paragraphe, une liste plus ou moins complexe, des images, un autre tableau, des blocs de code, etc.

## 17.2.4. Agir sur la largeur des colonnes

### 17.2.4.1. Comprendre la notion de largeur disponible

Pour agir sur la dimension des colonnes d'un tableau, il faut comprendre la notion d'espace horizontal disponible.

L'espace horizontal disponible est équivalent à 100% de l'élément qui contient le tableau (généralement, ce sera la largeur de la page dans le cas d'un fichier pdf).

### 17.2.4.2. Ajustement manuel de la largeur des colonnes

Avec `[cols="1,1,1"]`, il est indiqué que l'espace horizontal total occupé est de 3. Ce total correspond toujours à l'espace total disponible pour le tableau. Donc, si chaque colonne occupe un espace de 1, cela veut dire qu'elles occupent chacune un espace de  $1/3 = 33\%$ . Elles auront toutes la même largeur. S'il avait été indiqué `[cols="2,2,2"]`, les colonnes auraient toujours la même largeur puisque le total fait 6 et donc  $2/6 = 1/3$ .

Illustrons la gestion des largeurs de colonnes :

- exemple 1 : la première colonne occupe la moitié (2/4) de l'espace horizontal total, les autres, 1/4

```
[cols="2,1,1"]
```

A	B	C
---	---	---

- exemple 2 : la première colonne occupe  $2/7^{\text{ème}}$  de l'espace horizontal total, la seconde  $1/7^{\text{ème}}$  et la dernière  $4/7^{\text{ème}}$

```
[cols="2,1,4"]
```

A	B	C
---	---	---

Lorsque les colonnes ont toutes la même largeur, il n'est pas nécessaire d'écrire autant de 1 que de colonnes. Dans ce cas, le caractère \* peut être utilisé comme **multiplicateur de colonne**

```
[cols="3*"] ①
```

① le tableau sera constitué de 3 colonnes de largeur identique :

A	B	C
---	---	---

Les deux approches peuvent être mixées :

```
[cols="2,3*"] ①
```

① tableau de 4 colonnes dont les 3 dernières ont la même largeur et la première le double de cette largeur

A	B	C	D
---	---	---	---

La largeur des colonnes peut être exprimée en pourcentage dont le total doit être égal à 100% :



```
[cols="20%,30%,50%"]
|===
|A
|B
|C
|===
```

#### 17.2.4.3. Ajustement automatique en fonction du contenu

L'option `%autowidth` permet de spécifier que les colonnes doivent s'adapter à leur contenu. Le caractère `%` indique qu'il s'agit d'une option (c'est une syntaxe raccourcie).

```
%autowidth,cols="1,1"
|===
| A
| B
|
| C
| anticonstitutionnellement
|===
```

Comme vous pouvez le constater, le tableau n'occupe alors que l'espace nécessaire. Sachez qu'il est possible de [spécifier la largeur d'un tableau](#).

A	B
C	anticonstitutionnellement

Si vous souhaitez que le tableau occupe tout de même l'intégralité de l'espace tout en utilisant l'option `%autowidth`, il faut ajouter le rôle `.stretch` :

```
[%autowidth.stretch,cols="1,1"] ①  
|====  
| A  
| B  
  
| C  
| anticonstitutionnellement  
|====
```

① le rôle `.stretch` indique que le tableau doit s'étirer pour occuper tout l'espace disponible.

A	B
C	anticonstitutionnellement

Pour aller plus loin dans la définition des largeurs de colonnes, vous pouvez lire [cette page de documentation dédiée](#).

### 17.2.5. Agir sur la largeur du tableau

Un tableau va occuper tout l'espace horizontal dont il dispose (sauf lorsque l'option `%autowidth` est spécifiée).

Ce comportement par défaut peut être modifié en spécifiant explicitement la largeur que doit occuper le tableau (par rapport à son conteneur) avec l'attribut `width` :

```
[cols="20%,30%,50%", width=70%] ①  
|====  
|A  
|B  
|C  
|====
```

① l'attribut `width` permet de spécifier la largeur du tableau dans son conteneur.

Voici un tableau occupant 70% de la largeur de la page :

A	B	C
---	---	---

Voici le code AsciiDoc d'un tableau devant avoir une largeur égale à 50% de son conteneur. Le conteneur est ici le bloc d'avertissement.



```
[cols="50%,10%,40%", width=50%]  
|====  
|A  
|B
```

```
|C  
|==
```

Le tableau ci-dessous fait bien 50% de la largeur du bloc d'avertissement :

A	B	C
---	---	---

## 17.3. Ajouter un titre au tableau

### 17.3.1. Où écrire le titre du tableau ?

Ajouter un titre se fait exactement comme l'ajout d'un titre sur une image, une liste, un paragraphe. Il suffit de précéder le tableau du caractère `.` suivi du titre (il ne faut pas d'espace entre le point et le premier caractère du titre)

```
.tableau contenant des lettres de l'alphabet ①  
[cols="1,1,1"]  
|===  
|A  
|B  
|C  
|==
```

① titre du tableau

Table 3. tableau contenant des lettres de l'alphabet

A	B	C
---	---	---

### 17.3.2. Personnaliser l'étiquette du titre d'un tableau

Le titre d'un tableau est préfixé par le mot `Table` suivi d'un numéro d'apparition dans le document :

<code>Table 1.</code>	tableau contenant des lettres de l'alphabet	
A	B	C

Figure 65. étiquette de titre par défaut

Il est possible de personnaliser l'étiquette de titre avec l'attribut de document `:table_caption:` suivi du nom d'étiquette que vous souhaitez utiliser. Pour rappel, un attribut de document est un attribut qui permet de modifier le rendu du document : voir le chapitre [chapitre 19, Les attributs de document](#))

```
= titre principal du document  
:table-caption: titre du tableau ! ①
```

① personnalisation de l'étiquette affichée avant chaque titre de tableau pour tout le document.

<b>titre du tableau ! 2. tableau contenant des lettres de l'alphabet</b>		
A	B	C

Figure 66. étiquette de titre personnalisée

Si vous ne souhaitez pas utiliser les étiquettes de titre pour vos tableaux, vous pouvez les désactiver en annulant l'attribut `table-caption` avec le caractère ! :

= titre principal du document :!table-caption: ①
---

① le point d'exclamation annule l'attribut (c'est comme s'il n'existait plus). Il n'y aura plus d'étiquette de titre pour vos tableaux :

<b>X tableau contenant des lettres de l'alphabet</b>		
A	B	C

Figure 67. titre de tableau sans étiquette

## 17.4. Définir l'entête et le pied de tableau

### 17.4.1. Définir les entêtes de colonne

Pour indiquer que la première ligne correspond aux entêtes des colonnes, il faut spécifier l'option `header` avec le caractère % :

[%header, cols="2*a"] ①
==
titre de la colonne A
titre de la colonne B
du contenu
et encore du contenu
==

① la valeur d'option `header` indique que la première ligne correspond aux entêtes

titre de la colonne A	titre de la colonne B
du contenu	et encore du contenu

Figure 68. tableau avec entêtes de colonne



Il est possible de cumuler des options en les listant sans les séparer :

```
[%header%autowidth.stretch,cols="2*"] ①
| ===
| A
| titre de la colonne B

| du contenu
| et encore du contenu
| ===
```

① L'option d'entête de colonne et d'ajustement au contenu sont définies.

A	titre de la colonne B
du contenu	et encore du contenu mais plus long

Les cellules constituant les entêtes de colonne font l'objet d'une mise en forme spécifique (dans l'image, les titres sont en gras)

#### 17.4.2. Définir le pied de tableau

Un pied de tableau est parfois utile pour mettre en valeur la dernière ligne du tableau. Typiquement, une ligne de total est une ligne candidate pour devenir pied de tableau.

Pour spécifier que le tableau à un pied, il faut spécifier l'option `footer` :

```
[%autowidth%header%footer,cols="3*"] ① ②
| ===
| &nbsp;
| effectif 2023
| effectif 2025
// -----td3-----
| hommes
| 250
| 350
// -----td3-----
| femmes
| 180
| 320
// -----td3-----
| Total
| 430
| 670
| ===
```

① l'option `header` indique que le tableau à une ligne d'entête

② l'option `footer` indique le tableau à un ligne de pied de tableau

	<b>effectif 2023</b>	<b>effectif 2025</b>
hommes	250	350
femmes	180	320
Total	430	670

Figure 69. tableau avec une ligne d'entête et une ligne de pied de tableau

## 17.5. Aligner le contenu d'une colonne

### 17.5.1. Alignement horizontal au niveau d'une colonne

Pour un alignement horizontal du contenu de toutes les cellules d'une colonne, il faut utiliser un opérateur d'alignement.

Un **opérateur d'alignement dans AsciiDoc** est un caractère qui permet d'indiquer en langage AsciiDoc qu'il faut prévoir un alignement spécifique pour le contenu de toutes les cellules d'une colonne.

Table 4. opérateurs d'alignement horizontal disponibles

Opérateur d'alignement horizontal	Signification
<	alignement du contenu à gauche
>	alignement du contenu à droite
^	centrage du contenu



**L'opérateur d'alignement doit être placé à gauche du spécificateur de colonne (le nombre qui indique la largeur de la colonne)**



Par défaut, le contenu est aligné sur la gauche.

Voici un tableau dont le contenu de la première colonne sera centré horizontalement, le contenu de la seconde sera aligné à gauche et celui de la dernière colonne à droite.

```
[%header,cols="^1,<1,>1"] ① ② ③
|===
|
| effectif 2023
| effectif 2025
// -----td3-----
| hommes
```

```

| 250
| 350
// -----td3-----
| femmes
| 180
| 320
// -----td3-----
| enfants
| 58
| 85
|===

```

- ① `^1` le caractère `^` placé avant le spécificateur de colonne `1` indique que le contenu des cellules de la première colonne doit être centré horizontalement
- ② `<1` le caractère `<` placé avant le spécificateur de colonne `1` indique que le contenu des cellules de la seconde colonne doit être aligné à gauche
- ③ `>1` le caractère `>` placé avant le spécificateur de colonne `1` indique que le contenu des cellules de la seconde colonne doit être aligné à droite

	<b>effectif 2023</b>	<b>effectif 2025</b>
hommes	250	350
femmes	180	320
enfants	58	85

Figure 70. tableau avec alignement du contenu par colonne

Si les colonnes sont spécifiées à l'aide du multiplicateur `*`, l'opérateur d'alignement doit être placé à sa droite :

```

[%header,cols="3*^"] ①
|===
|
| effectif 2023
| effectif 2025
// -----td3-----
| hommes
| 250
| 350
// -----td3-----
| femmes
| 180
| 320
// -----td3-----
| enfants
| 58
| 85
|===

```

① L'opérateur d'alignement est placé après le multiplicateur `*`

	effectif 2023	effectif 2025
hommes	250	350
femmes	180	320
enfants	58	85

Figure 71. tableau dont les colonnes ont un contenu centré horizontalement

L'alignement horizontal peut être combiné avec l'alignement vertical.

### 17.5.2. Alignement vertical au niveau d'une colonne

Il faut avoir compris l'[alignement horizontal sur une colonne](#) pour bien comprendre l'alignement vertical.

L'objectif est d'avoir un contenu centré verticalement dans les cellules d'une colonne ou de placer le contenu en haut ou en bas des cellules.

Il faut spécifier un opérateur d'alignement en fonction de l'alignement vertical désiré (centrage vertical, en bas, en haut). Les opérateurs sont les mêmes que les [opérateurs d'alignement horizontal](#) à la différence qu'ils sont préfixés d'un point `.` :

Table 5. opérateurs d'alignement vertical disponibles

Opérateur d'alignement vertical	Signification
<code>.&lt;</code>	alignement du contenu en haut
<code>.&gt;</code>	alignement du contenu en bas
<code>.^</code>	centrage vertical du contenu



Par défaut, le contenu d'une cellule est aligné sur le haut.



L'opérateur d'alignement vertical doit être spécifié **après l'opérateur d'alignement horizontal et avant le spécificateur de colonne**.

```
[%header,cols=".^1,.<1,.>1"] ①
=====
|
| effectif 2023
| effectif 2025
// -----td3-----
```

```

| catégorie des +
moins de 60kg
| 250
| 350
// -----td3-----
| catégorie des +
moins de 80kg
| 180
| 320
// -----td3-----
| catégorie des +
moins de 100kg
| 58
| 85
|===

```

- ① le contenu de la première colonne est centré verticalement `.^`, le contenu de la seconde colonne est aligné sur le haut  et la troisième colonne a son contenu aligné sur le bas `.>`.

	<b>effectif 2023</b>	<b>effectif 2025</b>
catégorie des moins de 60kg	250	350
catégorie des moins de 80kg	180	320
catégorie des moins de 100kg	58	85

Figure 72. tableau avec alignement vertical du contenu par colonne

### 17.5.3. Combiner alignement horizontal et vertical au niveau d'une colonne

Il faut bien comprendre l'**alignement horizontal** et l'**alignement vertical** pour combiner les deux.



L'opérateur d'alignement horizontal doit être spécifié **avant le spécificateur de colonne** et **avant l'opérateur d'alignement vertical**



L'opérateur d'alignement vertical doit être spécifié **après l'opérateur d'alignement horizontal** et **avant le spécificateur de colonne**.

Voici un tableau dont le contenu des cellules de la première colonne va être centré à droite et au milieu (vertical) , celui de la seconde colonne centré horizontalement et verticalement et celui de la dernière colonne aligné à gauche et en bas :

```

[%header,cols=".^.^.1,<.>1"]
|===
|
```

```

| effectif 2023
| effectif 2025
// -----td3-----
| catégorie des +
moins de 60kg

| 250
| 350
// -----td3-----
| catégorie des +
moins de 80kg
| 180
| 320
// -----td3-----
| catégorie des +
moins de 100kg
| 58
| 85
| ===

```

	<b>effectif 2023</b>	<b>effectif 2025</b>
catégorie des moins de 60kg	250	350
catégorie des moins de 80kg	180	320
catégorie des moins de 100kg	58	85

Figure 73. tableau combinant alignements horizontal et vertical

## 17.6. Aligner le contenu au niveau d'une cellule

Nous avons abordé l'[alignement horizontal au niveau d'une colonne](#) et l'[alignement vertical au niveau d'une colonne](#).

Parfois, vous pouvez avoir besoin d'agir au niveau d'une cellule bien précise. Cela est possible avec AsciiDoc.

Si vous connaissez déjà les [opérateurs d'alignement horizontal](#) et les [opérateurs d'alignement vertical](#), alors il va vous être très simple de régler l'alignement au niveau d'une cellule.

Pour gérer l'alignement au niveau d'une cellule, il suffit d'utiliser les opérateurs d'alignement **avant le marqueur de cellule** | :

```

.tableau dont le contenu des cellules
[%awidth%header,cols="3**"]
|===
| colonne A

```

```
| colonne B
| colonne C
```

// -----td2-----

```
<| un contenu + ①
aligné à gauche +
et qui s'étale +
sur plusieurs lignes
| contenu aligné en haut et à gauche
```

(comportement par défaut)

```
^.>| contenu aligné en bas mais centré ②
```

// -----td2-----

```
>| un contenu + ③
aligné à droite +
et qui s'étale +
sur plusieurs lignes
```

```
>.>| contenu aligné + ④
```

à droite et en bas

```
>.^| contenu centré verticalement et à droite ⑤
```

|---

① le caractère < spécifié avant le marqueur de cellule | indique un alignement à gauche

② les caractères ^.> spécifiés indiquent un alignement centré horizontalement et aligner en bas verticalement

③ les caractères > spécifiés indiquent un alignement horizontal à droite

④ les caractères >.> spécifiés indiquent un alignement horizontal à droite et verticalement en bas

⑤ les caractères >.^ spécifiés indiquent un alignement horizontal à droite et centré verticalement

colonne A	colonne B	colonne C
un contenu aligné à gauche et qui s'étale sur plusieurs lignes	contenu aligné en haut et à gauche (comportement par défaut)	contenu aligné en bas mais centré
un contenu aligné à droite et qui s'étale sur plusieurs lignes		contenu centré verticalement et à droite contenu aligné à droite et en bas

Figure 74. tableau montrant un alignement horizontal et vertical par cellule



Dès lors que vos tableaux ont une ligne d'entête, prenez l'habitude de centrer verticalement et horizontalement le contenu au niveau de chacune des cellules

d'entête (sauf si ce rendu ne vous convient pas).

```
[%autowidth%header,cols="3*"]
|===
^.^| entête A ①
^.^| entête B ①
^.^| entête C ①

// lignes du tableau...
|===
```

① ^.^ centrage horizontal et vertical du contenu de la cellule

Cela peut paraître fastidieux à écrire mais avec les lives template, cela peut être automatisé (voir le chapitre sur les lives template)

## 17.7. Centrer le tableau dans la page

Un tableau qui n'occuperait pas toute la largeur disponible est placé par défaut à gauche.

Pour **centrer un tableau** dans la page, il faut utiliser l'attribut `align` et la valeur d'alignement horizontale `center`. D'autres valeurs sont disponibles : `left` (par défaut) et `right`.

```
.ombrage des lignes avec `stripes=even`
[%autowidth%header,cols="1,^1m,^1m",stripes=even,width=50%,align="center"] ①
|===
^.^| référence
^.^| prix
^.^| quantité
// -----td2-----
| 48754FDR
| 145.21
| 235.00

// -----td2-----
| 85974ED
| 500.00
| 750.00
// -----td2-----

| 325471J
| 212.30
| 174.23
|===
```

① L'attribut `align` permet de spécifier un alignement horizontal. Ici, le tableau sera centré dans la page.

	référence	prix	quantité
	48754FDR	145.21	235.00
	85974ED	500.00	750.00
	325471J	212.30	174.23

Figure 75. centrage d'un tableau avec l'attribut `align=center`



La prévisualisation générée par le plugin ne prend pas en charge le centrage. Toutefois, le tableau sera centré dans le fichier pdf de sortie.

## 17.8. Mise en forme du contenu au niveau d'une colonne

Tout comme il est possible de gérer l'alignement du contenu d'une colonne, il est possible de gérer la mise en forme (basique) de toute les cellule d'une colonne.

Pour appliquer une mise en forme au niveau d'une colonne, il faut utiliser un opérateur de style.

Un **opérateur de style** est un caractère qui permet d'indiquer en langage AsciiDoc qu'il faut prévoir une mise en forme spécifique pour toutes les cellules d'une colonne.



L'opérateur de style doit être spécifié **après le spécificateur de colonne**.



Par défaut, seuls les éléments de mise en forme typographique sont fonctionnels dans une cellule de tableau (mise en gras, en italique, surlignage, etc.).

Les listes, images, tableau, etc. seront affichés comme s'il s'agissait de texte simple.

Voici un tableau des 6 styles applicables à une colonne :

Opérateur de style	Signification
d	(d pour default) c'est le rendu par défaut dans une cellule. Ne rien spécifier revient à spécifier d
s	(s pour strong) met en gras le contenu
a	(a pour AsciiDoc) permet d'utiliser dans une cellule des blocs de code, des images, des listes, d'autres tableaux, etc. <b>C'est de loin l'opérateur le plus utile</b>
e	(e pour emphasis) permet de mettre en italique le contenu de la colonne

Opérateur de style	Signification
h	(h pour header) permet d'appliquer le style d'entête aux cellules de la colonne (utile pour un tableau à double entrées)
l	(l pour literal) le contenu sera traité comme un bloc littéral (ce qui est écrit est rendu à l'identique : retour à la ligne, caractère de mise en forme, etc.)

Voici un tableau qui utilise l'opérateur de style h pour la première colonne et m pour la seconde et s pour la troisième :

```
[%awidth%header,cols="1h,1m,1l"] ①
|===
|^ | référence
|^ | prix
|^ | quantité
// -----td2-----
| 48754FDR
| 145.21
| 235.00

// -----td2-----

| 85974ED
| 500
| 750
|===
```

① le caractère h pour "header" permet de créer des entêtes de ligne

référence	prix	quantité
48754FDR	145.21	235.00
85974ED	500	750

Figure 76. tableau utilisant des opérateurs de style sur ses colonnes

Pour bien comprendre les différents opérateurs de style, voici le rendu d'une cellule avec le même contenu. **Seul l'opérateur de style change à chaque fois.**

Voici le contenu utilisé pour chaque exemple de rendu :

```
[%awidth%header,cols="1"]
|===
|^ | Rendu sans aucun opérateur de style
// -----td1-----
|
```

\*Quand Chuck Norris\* \*code\*, \_l'ordinateur\_ \*s'auto-corrigé\*. Si ça ne marche toujours pas, il utilise les [.underline]#astuces suivantes# :

- \* \*Redémarrer\* l'ordinateur par la seule force de son regard.
  - \* Surligner les lignes de code #défectueuses# avec un marqueur \_\_invisible\_\_.
  - \* Ajouter une note de bas de page expliquant que l'[.line-through]#erreur# \*n'existe pas\*.
- |==

## Pas d'opérateur de style

```
[%awidth%header,cols="1"]  
//...
```

### Rendu sans aucun opérateur de style

**Quand Chuck Norris code, l'ordinateur s'auto-corrigé.** Si ça ne marche toujours pas, il utilise les astuces suivantes :

\* Redémarrer l'ordinateur par la seule force de son regard. \* Surligner les lignes de code défectueuses avec un marqueur invisible. \* Ajouter une note de bas de page expliquant que l'erreur n'existe pas.

Figure 77. Rendu sans opérateur de style appliqué sur la colonne



Sans opérateur de style, seuls les éléments de mise en forme de texte sont rendus. La liste reste une simple chaîne de caractères.

## Utilisation de l'opérateur de style **d** pour "default"

```
[%awidth%header,cols="1d"]  
//...
```

### Rendu avec opérateur de style **d** (comportement par défaut)

**Quand Chuck Norris code, l'ordinateur s'auto-corrigé.** Si ça ne marche toujours pas, il utilise les astuces suivantes :

\* Redémarrer l'ordinateur par la seule force de son regard. \* Surligner les lignes de code défectueuses avec un marqueur invisible. \* Ajouter une note de bas de page expliquant que l'erreur n'existe pas.

Figure 78. Rendu avec opérateur de style **d** pour "default"



L'opérateur de style **d** revient à ne pas définir d'opérateur de style. Seuls les éléments

de mise en forme de texte sont rendus. La liste reste une simple chaîne de caractères.

### Utilisation de l'opérateur de style **s** pour "strong"

```
[%awidth%header,cols="1s"]  
//...
```

#### Rendu avec opérateur de style **s** (en gras)

Quand Chuck Norris code, l'ordinateur s'auto-corrigé. Si ça ne marche toujours pas, il utilise les astuces suivantes :

\* Redémarrer l'ordinateur par la seule force de son regard. \* Surligner les lignes de code **défectueuses** avec un marqueur *invisible*. \* Ajouter une note de bas de page expliquant que l'**erreur** n'existe pas.

Figure 79. Rendu avec opérateur de style **s** pour "strong"

L'opérateur de style **s** met tout le contenu en gras (en dehors des éléments en italique semble-t-il)

La mise en forme typographique est bien rendue mais la liste n'est pas traitée comme telle.

### Utilisation de l'opérateur de style **a** pour "asciidoc"

```
[%awidth%header,cols="1a"]  
//...
```

#### Rendu avec opérateur de style **a** (AsciiDoc)

Quand Chuck Norris code, l'ordinateur s'auto-corrigé. Si ça ne marche toujours pas, il utilise les astuces suivantes :

- Redémarrer l'ordinateur par la seule force de son regard.
- Surligner les lignes de code **défectueuses** avec un marqueur *invisible*.
- Ajouter une note de bas de page expliquant que l'**erreur** n'existe pas.

Figure 80. Rendu avec opérateur de style **a** pour default

L'opérateur de style **a** est pratiquement toujours utilisé car les cellules d'une colonne sont susceptibles de contenir des éléments de type liste, image, etc.

La mise en forme typographique est bien rendue ainsi que la liste.

 Je vous conseille de toujours spécifier l'opérateur de style **a** sur vos colonnes. La

création d'un live template permet d'automatiser cela. (voir le chapitre dédié aux lives template)

## Utilisation de l'opérateur de style **h** pour "header"

```
[%awidth%header,cols="1h"]  
//...
```

### Rendu avec opérateur de style **h** (entête)

Quand Chuck Norris code, l'ordinateur s'auto-corrigé. Si ça ne marche toujours pas, il utilise les astuces suivantes :

\* Redémarrer l'ordinateur par la seule force de son regard. \* Surligner les lignes de code défectueuses avec un marqueur invisible. \* Ajouter une note de bas de page expliquant que l'erreur n'existe pas.

Figure 81. Rendu avec opérateur de style **h** pour "header"



L'opérateur de style **h** formate les cellules à l'identique d'une cellule d'entête.

La mise en forme typographique est rendue mais pas la liste.

## Utilisation de l'opérateur de style **e** pour "emphasis"

```
[%awidth%header,cols="1e"]  
//...
```

### Rendu avec opérateur de style **e** (italique)

Quand Chuck Norris code, l'ordinateur s'auto-corrigé. Si ça ne marche toujours pas, il utilise les astuces suivantes :

\* Redémarrer l'ordinateur par la seule force de son regard. \* Surligner les lignes de code défectueuses avec un marqueur invisible. \* Ajouter une note de bas de page expliquant que l'erreur n'existe pas.

Figure 82. Rendu avec opérateur de style **e** pour "emphasis"



L'opérateur de style **e** met le contenu de la cellule en italique (sauf les contenus en gras).

La mise en forme typographique est bien rendue mais pas la liste.

## Utilisation de l'opérateur de style **l** pour "litteral"

```
[%autowidth%header,cols="11"]  
//...
```

### Rendu avec opérateur de style `l` (littéral)

\*Quand Chuck Norris\* \*code\*, \_l'ordinateur\_ \*s'auto-corrigé\*. Si ça ne marche toujours pas, il utilise les [.underline]#astuces suivantes# :

- \* \*Redémarrer\* l'ordinateur par la seule force de son regard.
- \* Surligner les lignes de code #défectueuses# avec un marqueur \_\_invisible\_\_.
- \* Ajouter une note de bas de page expliquant que l'[.line-through]#erreur# \*n'existe pas\*.

Figure 83. Rendu avec opérateur de style `l` pour default



L'opérateur de style `l` traite le contenu de la cellule comme un contenu littéral. Ce qui est écrit est rendu tel qu'il est écrit.

Aucune mise en forme n'est appliquée.



Il ne peut y avoir qu'un seul opérateur de style par colonne.

## 17.9. Mise en forme au niveau du contenu d'une cellule.

La mise en forme au niveau du contenu d'une cellule fonctionne comme l'alignement au niveau d'une cellule.

Encore une fois, il s'agit d'une mise en forme basique.

L'[opérateur de style](#) doit être spécifié au niveau de la cellule, avant le marqueur de cellule `|`.

```
[cols="1,1,2"]  
|===  
| référence  
h| couleurs disponibles  
e| quantité  
// -----td2-----  
s| 48754FDR  
a| * rouge  
* vert  
* bleu  
| * jaune  
* blanc  
* violet  
  
// -----td2-----
```

```
m| 85974ED
d| 500
l| 750 (par lot de 2)
|==
```

référence	couleurs disponibles	quantité
<b>48754FDR</b>	<ul style="list-style-type: none"> <li>• rouge</li> <li>• vert</li> <li>• bleu</li> </ul>	* jaune * blanc * violet
<b>85974ED</b>	500	750 (par lot de 2)

Figure 84. mise en forme au niveau du contenu des cellules

Les opérateurs de style peuvent être utilisés avec les opérateurs d'alignement :

```
[cols="1,1,2"]
|===
^.^| référence
^.^h| couleurs disponibles
^.^e| quantité
// -----td2-----
s| 48754FDR
a| * rouge
* vert
* bleu
>.>| * jaune
* blanc
* violet

// -----td2-----

>m| 85974ED +
745863
^.>d| 500
l| 750 (par lot de 2)
|==
```

référence	couleurs disponibles	quantité
<b>48754FDR</b>	<ul style="list-style-type: none"> <li>• rouge</li> <li>• vert</li> <li>• bleu</li> </ul>	* jaune * blanc * violet
<b>85974ED</b> <b>745863</b>	500	750 (par lot de 2)

Figure 85. cumul de l'alignement et de la mise en forme au niveau des cellules

## 17.10. Les bordures de tableau

### 17.10.1. Les bordures qui entourent le tableau

La **bordure d'un tableau** peut être spécifiée avec l'attribut `frame` tel que :

```
[%autowidth%header,cols="1,^1m,^1m",width=50%,frame=ends] ①
|===
//ici les lignes du tableau
|==
```

- ① plusieurs attributs sont utilisés (`cols`, `width`, `frame`). L'attribut `frame` reçoit une valeur qui peut être une des suivantes : `all`, `ends`, `sides` ou `none`.

*Table 6. valeurs de l'attribut frame pour les bordures de tableau*

Valeur d'attribut <code>frame</code>	Effet									
<code>all</code>	<p>bordure tracée sur les 4 côtés du tableau (comportement par défaut)</p> <table border="1"> <thead> <tr> <th>référence</th><th>prix</th><th>quantité</th></tr> </thead> <tbody> <tr> <td>48754FDR</td><td>145.21</td><td>235.00</td></tr> <tr> <td>85974ED</td><td>500.00</td><td>750.00</td></tr> </tbody> </table> <p><i>Figure 86. bordures de tableau définies avec <code>frame=all</code></i></p>	référence	prix	quantité	48754FDR	145.21	235.00	85974ED	500.00	750.00
référence	prix	quantité								
48754FDR	145.21	235.00								
85974ED	500.00	750.00								
<code>ends</code>	<p>bordures haute et basse seulement</p> <table border="1"> <thead> <tr> <th>référence</th><th>prix</th><th>quantité</th></tr> </thead> <tbody> <tr> <td>48754FDR</td><td>145.21</td><td>235.00</td></tr> <tr> <td>85974ED</td><td>500.00</td><td>750.00</td></tr> </tbody> </table> <p><i>Figure 87. bordures de tableau définies avec <code>frame=ends</code></i></p>	référence	prix	quantité	48754FDR	145.21	235.00	85974ED	500.00	750.00
référence	prix	quantité								
48754FDR	145.21	235.00								
85974ED	500.00	750.00								
<code>sides</code>	<p>bordures gauche et droite seulement</p> <table border="1"> <thead> <tr> <th>référence</th><th>prix</th><th>quantité</th></tr> </thead> <tbody> <tr> <td>48754FDR</td><td>145.21</td><td>235.00</td></tr> <tr> <td>85974ED</td><td>500.00</td><td>750.00</td></tr> </tbody> </table> <p><i>Figure 88. bordures de tableau définies avec <code>frame=sides</code></i></p>	référence	prix	quantité	48754FDR	145.21	235.00	85974ED	500.00	750.00
référence	prix	quantité								
48754FDR	145.21	235.00								
85974ED	500.00	750.00								

Valeur d'attribut frame	Effet									
none	<p>pas de bordure</p> <table border="1"> <thead> <tr> <th>référence</th> <th>prix</th> <th>quantité</th> </tr> </thead> <tbody> <tr> <td>48754FDR</td> <td>145.21</td> <td>235.00</td> </tr> <tr> <td>85974ED</td> <td>500.00</td> <td>750.00</td> </tr> </tbody> </table>	référence	prix	quantité	48754FDR	145.21	235.00	85974ED	500.00	750.00
référence	prix	quantité								
48754FDR	145.21	235.00								
85974ED	500.00	750.00								

Figure 89. bordures de tableau définies avec `frame=none`

Si vous utilisez un style de bordure de tableau différent du style par défaut pour tous vos tableaux, vous pouvez définir le comportement par défaut à l'aide de l'attribut de document `table-frame` avec la valeur `all`, `ends`, `sides` ou `none` :



= titre principal du document  
`:table-frame: ends` ①

① tous les tableaux adopteront par défaut le style associé à la valeur `ends`.

Le style d'un tableau en particulier pourra toujours être modifié à l'aide de l'attribut `frame`.

### 17.10.2. Les bordures entre les cellules

Les bordures entre les cellules peuvent être configurées avec l'attribut `grid` qui accepte une des valeurs suivantes : `all`, `rows`, `cols` ou `none`.

Voici un exemple d'utilisation de l'attribut `grid` :

```
[%autowidth%header,cols="1,^1m,^1m",width=50%,grid=cols] ①
|===
//ici les lignes du tableau
|==
```

① plusieurs attributs sont utilisés (`cols`, `width`, `grid`). L'attribut `grid` reçoit une valeur qui peut être une des suivantes : `all`, `rows`, `cols` ou `none`.

Table 7. valeurs de l'attribut grid pour les bordures entre cellules

Valeur d'attribut grid	Effet									
all	bordure tracée sur les 4 côtés du tableau (comportement par défaut)									
	<table border="1"> <thead> <tr> <th>référence</th><th>prix</th><th>quantité</th></tr> </thead> <tbody> <tr> <td>48754FDR</td><td>145.21</td><td>235.00</td></tr> <tr> <td>85974ED</td><td>500.00</td><td>750.00</td></tr> </tbody> </table> <p><i>Figure 90. bordures de tableau définies avec grid=all</i></p>	référence	prix	quantité	48754FDR	145.21	235.00	85974ED	500.00	750.00
référence	prix	quantité								
48754FDR	145.21	235.00								
85974ED	500.00	750.00								
rows	les traits verticaux entre les cellules ne sont pas tracés									
	<table border="1"> <thead> <tr> <th>référence</th><th>prix</th><th>quantité</th></tr> </thead> <tbody> <tr> <td>48754FDR</td><td>145.21</td><td>235.00</td></tr> <tr> <td>85974ED</td><td>500.00</td><td>750.00</td></tr> </tbody> </table> <p><i>Figure 91. bordures de tableau définies avec grid=rows</i></p>	référence	prix	quantité	48754FDR	145.21	235.00	85974ED	500.00	750.00
référence	prix	quantité								
48754FDR	145.21	235.00								
85974ED	500.00	750.00								
cols	les traits horizontaux entre les cellules ne sont pas tracés									
	<table border="1"> <thead> <tr> <th>référence</th><th>prix</th><th>quantité</th></tr> </thead> <tbody> <tr> <td>48754FDR</td><td>145.21</td><td>235.00</td></tr> <tr> <td>85974ED</td><td>500.00</td><td>750.00</td></tr> </tbody> </table> <p><i>Figure 92. bordures de tableau définies avec grid=cols</i></p>	référence	prix	quantité	48754FDR	145.21	235.00	85974ED	500.00	750.00
référence	prix	quantité								
48754FDR	145.21	235.00								
85974ED	500.00	750.00								
none	pas de bordure entre les cellules									
	<table border="1"> <thead> <tr> <th>référence</th><th>prix</th><th>quantité</th></tr> </thead> <tbody> <tr> <td>48754FDR</td><td>145.21</td><td>235.00</td></tr> <tr> <td>85974ED</td><td>500.00</td><td>750.00</td></tr> </tbody> </table> <p><i>Figure 93. bordures de tableau définies avec grid=none</i></p>	référence	prix	quantité	48754FDR	145.21	235.00	85974ED	500.00	750.00
référence	prix	quantité								
48754FDR	145.21	235.00								
85974ED	500.00	750.00								

Si vous utilisez un style de bordure entre les cellules différent du style par défaut pour tous vos tableaux, vous pouvez définir le comportement par défaut à l'aide de l'attribut de document `table-grid` avec la valeur `all`, `rows`, `cols` ou `none` :

= titre principal du document  
`:table-grid: rows ①`

① tous les tableaux adopteront par défaut le style associé à la valeur `rows`.

Le style d'un tableau en particulier peut toujours être modifié à l'aide de l'attribut `grid`.

## 17.11. Ombrer une ligne sur deux

Pour alterner entre ligne ombrée et non ombrée, l'attribut `stripes` peut être utilisé avec une des valeurs du tableau ci-après :

Table 8. valeurs de l'attribut `grid` pour les bordures entre cellules

Valeur d'attribut <code>stripes</code>	Effet												
<code>none</code>	<p>aucune ligne n'est ombrée (comportement par défaut)</p> <table border="1"> <thead> <tr> <th>référence</th><th>prix</th><th>quantité</th></tr> </thead> <tbody> <tr> <td>48754FDR</td><td>145.21</td><td>235.00</td></tr> <tr> <td>85974ED</td><td>500.00</td><td>750.00</td></tr> <tr> <td>325471J</td><td>212.30</td><td>174.23</td></tr> </tbody> </table>	référence	prix	quantité	48754FDR	145.21	235.00	85974ED	500.00	750.00	325471J	212.30	174.23
référence	prix	quantité											
48754FDR	145.21	235.00											
85974ED	500.00	750.00											
325471J	212.30	174.23											
<code>even</code>	<p>les lignes <b>paires</b> sont ombrées</p> <table border="1"> <thead> <tr> <th>référence</th><th>prix</th><th>quantité</th></tr> </thead> <tbody> <tr> <td>48754FDR</td><td>145.21</td><td>235.00</td></tr> <tr> <td>85974ED</td><td>500.00</td><td>750.00</td></tr> <tr> <td>325471J</td><td>212.30</td><td>174.23</td></tr> </tbody> </table> <p>← ligne ombrée</p>	référence	prix	quantité	48754FDR	145.21	235.00	85974ED	500.00	750.00	325471J	212.30	174.23
référence	prix	quantité											
48754FDR	145.21	235.00											
85974ED	500.00	750.00											
325471J	212.30	174.23											
<code>odd</code>	<p>les lignes <b>impaires</b> sont ombrées</p> <table border="1"> <thead> <tr> <th>référence</th><th>prix</th><th>quantité</th></tr> </thead> <tbody> <tr> <td>48754FDR</td><td>145.21</td><td>235.00</td></tr> <tr> <td>85974ED</td><td>500.00</td><td>750.00</td></tr> <tr> <td>325471J</td><td>212.30</td><td>174.23</td></tr> </tbody> </table> <p>lignes ombrées</p>	référence	prix	quantité	48754FDR	145.21	235.00	85974ED	500.00	750.00	325471J	212.30	174.23
référence	prix	quantité											
48754FDR	145.21	235.00											
85974ED	500.00	750.00											
325471J	212.30	174.23											

Figure 94. ombrage des lignes avec `stripes=none`

Figure 95. ombrage des lignes avec `stripes=even`

Figure 96. ombrage des lignes avec `stripes=odd`

Valeur d'attribut <code>stripes</code>	Effet
<code>all</code>	toutes les lignes sont ombrées 

*Figure 97. ombrage des lignes avec stripes=all*

Si vous utilisez un ombrage de ligne différent du style par défaut pour tous vos tableaux, vous pouvez définir le comportement par défaut à l'aide de l'attribut de document `table-stripes` avec la valeur `none`, `even`, `odd` ou `all` :

= titre principal du document  
`:table-stripes: odd ①`

① tous les tableaux adopteront par défaut l'ombrage associé à la valeur `odd`.

Le style d'un tableau en particulier peut toujours être modifié à l'aide de l'attribut `stripes`.

## 17.12. Créer un tableau à partir de données au format CSV

### 17.12.1. Générer un tableau à partir de données csv

Le format csv pour « Comma Separated Values » permet de stocker des données dans un fichier texte. Les données sont séparées par une virgule, un point virgule ou tout autre délimiteur défini comme tel. Chaque retour à la ligne marque une nouvelle ligne de données.

Si vous n'avez jamais manipulé de fichier au format `csv`, pas de soucis ! Créez un nouveau fichier dans un tableur (Excel ou Calc par exemple)

	A	B	C
1	<b>Compétences en clin d'oeil</b>	<b>Compétences en fronçage des yeux</b>	<b>Compétences avec ses poils de jambes</b>
2	Eteindre un feu	Arrêter un train en marche	Tisser un pull en une seconde
3	Ecrire un roman	Faire fondre de la glace	Fabriquer un filet de pêche
4	Changer la direction du vent	Faire taire une salle bruyante	Broder une nappe en or
5	Décoder un message secret	Faire disparaître un nuage	Créer un parachute

*Figure 98. tableau de compétences de Chuck Norris*

Puis **enregistrez le fichier au format csv**. Fermez le fichier ouvert dans votre tableur et ouvrez-le

dans votre IDE :

1	Compétences en clin d'oeil;Compétences en fronçage des yeux;Compétences avec ses poils de jambes
2	Eteindre un feu;Arrêter un train en marche;Tisser un pull en une seconde
3	Ecrire un roman;Faire fondre de la glace;Fabriquer un filet de pêche
4	Changer la direction du vent;Faire taire une salle bruyante;Broder une nappe en or
5	Décoder un message secret;Faire disparaître un nuage;Créer un parachute
6	

Figure 99. fichier csv ouvert dans l'IDE

Les **données sont séparées par un point-virgule** (dans mon cas) et chaque ligne du tableau est une ligne de texte dans le fichier csv.

Pour créer un tableau à partir de données csv, il suffit de déclarer votre tableau en précisant le format source avec l'attribut `format` et la valeur `csv` :

```
[%header,format=csv]
|===
Compétences en clin d'oeil;Compétences en fronçage des yeux;Compétences avec ses poils de jambes
Eteindre un feu;Arrêter un train en marche;Tisser un pull en une seconde
Ecrire un roman;Faire fondre de la glace;Fabriquer un filet de pêche
Changer la direction du vent;Faire taire une salle bruyante;Broder une nappe en or
Décoder un message secret;Faire disparaître un nuage;Créer un parachute
|==
```

Cependant, le résultat n'est pas celui attendu :

Compétences en clin d'oeil;Compétences en fronçage des yeux;Compétences avec ses poils de jambes
Eteindre un feu;Arrêter un train en marche;Tisser un pull en une seconde
Ecrire un roman;Faire fondre de la glace;Fabriquer un filet de pêche
Changer la direction du vent;Faire taire une salle bruyante;Broder une nappe en or
Décoder un message secret;Faire disparaître un nuage;Créer un parachute

Figure 100. tentative de création d'un tableau à partir de données CSV avec séparateur point-virgule.

Effectivement, le séparateur de données ou délimiteur qui est pris en charge par défaut par AsciiDoc est la virgule. Donc, si nous avions défini au moment d'enregistrer notre fichier tableau au format csv le délimiteur virgule, notre tableau serait correctement rendu.

Mais, je ne l'ai pas fait volontairement pour traiter le cas qui se présente à nous.

Nous savons maintenant que le délimiteur traité par AsciiDoc avec le format csv est la virgule. Il faut lui « dire » d'utiliser le point-virgule. Cela peut être spécifié à l'aide de l'attribut `separator` et la valeur `;` :

```
[%header,cols="1,1,1",format=csv,separator=;] ①
|===
// données csv séparées par un point virgule
|==
```

① l'attribut `separator` permet de spécifier le délimiteur utilisé dans le fichier csv, ici le point virgule.

Compétences en clin d'oeil	Compétences en fronçage des yeux	Compétences avec ses poils de jambes
Eteindre un feu	Arrêter un train en marche	Tisser un pull en une seconde
Ecrire un roman	Faire fondre de la glace	Fabriquer un filet de pêche
Changer la direction du vent	Faire taire une salle bruyante	Broder une nappe en or
Décoder un message secret	Faire disparaître un nuage	Créer un parachute

Figure 101. tableau généré à partir de données csv et un séparateur personnalisé ;

Si vous voulez agir sur les colonnes (largeur, alignement, style), vous pouvez utiliser le spécificateur de colonnes :



```
[%header,cols="1,^1,1m",format=csv,separator=;] ①
|===
// ici des données csv séparées par un point virgule
|==
```

① des opérateurs d'alignement et de style sont mobilisés sur les colonnes pour agir sur le rendu du tableau

Cette fonctionnalité est déjà géniale quand on manipule des données csv. Mais le top du top, c'est que l'on peut inclure directement le fichier csv sans avoir à copier-coller les données csv dans le fichier AsciiDoc (voir le point suivant).

### 17.12.2. Générer un tableau à partir d'un fichier csv

Nous avons vu qu'il était possible de créer un tableau à partir de données csv. Celles-ci sont écrites directement dans le fichier AsciiDoc dans un bloc de code.

Le problème avec cette méthode, c'est que si les données sources sont mises à jour, il faut penser à copier-coller les nouvelles données dans le fichier AsciiDoc.

Heureusement, il est possible d'utiliser directement le fichier csv via une inclusion. La macro `include::[]` permet d'inclure un fichier.

Il faut spécifier le chemin relatif du fichier csv et terminer celui-ci par des crochets :

```
[%header,format=csv,separator=;]
```

```
|===
include::extras/competences_chuck_norris_2.csv[] ①
|==
```

① la macro `include::[]` permet d'inclure un fichier, c'est-à-dire que son contenu est tout simplement injecté dans le fichier AsciiDoc

Compétences en clin d'oeil	Compétences en fronçage des yeux	Compétences avec ses poils de jambes
Eteindre un feu	Arrêter un train en marche	Tisser un pull en une seconde
Ecrire un roman	Faire fondre de la glace	Fabriquer un filet de pêche
Changer la direction du vent	Faire taire une salle bruyante	Broder une nappe en or
Décoder un message secret	Faire disparaître un nuage	Créer un parachute

Figure 102. tableau généré à partir du fichier csv inclus depuis le fichier AsciiDoc

Pour en savoir plus à propos de l'utilisation de données permettant de générer des tableaux, vous pouvez lire [cette page de la documentation AsciiDoc](#).

## 17.13. D'autres fonctionnalités sur les tableaux

Ce chapitre aborde de nombreuses fonctionnalités du langage AsciiDoc concernant les tableaux, mais il n'en fait pas le tour !

Pour en faire davantage avec les tableaux, vous pouvez compléter vos connaissances des sujets suivants :

- [fusionner des cellules](#) horizontalement ou verticalement
- [cloner le contenu d'une cellule vers une autre cellule](#)
- [modifier l'orientation du tableau](#)
- [un tableau dans une cellule de tableau](#)
- [liste des attributs](#) relatifs à un tableau

# 18. Modifier l'orientation du document ou d'une page

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 20:54 | Auteur : Emmanuel Ravrat

Durée de réalisation : 15min

Il y a deux modes d'affichage :

- le mode portrait



- le mode paysage



En principe, les pages d'un fichier pdf sont affichées en mode "portrait". Cependant, il arrive qu'il soit nécessaire de changer l'orientation d'une ou de plusieurs pages dans le document final.

C'est le cas lorsque :

- une image est trop large
- un tableau est trop large
- et d'autres raisons que Chuck ne veut pas me donner...

A partir de maintenant, c'en est terminé de la création du fameux "saut de page" avec la gestion d'un point de section qui consiste à modifier le mode d'affichage. Oui, je fais référence à Word...

AsciiDoc rend cela **très** facile.

Lorsque dans votre document, vous arrivez à un contenu qui doit être affiché en mode "paysage" (*landscape* en anglais), vous créez un saut de page avec <<< et vous ajoutez le rôle **landscape** sur

celui-ci :

Cela donne le code AsciiDoc suivant :

```
//du contenu tout plein ici !  
  
[role=landscape]  
<<<  
  
// du contenu qui s'affiche au format paysage
```

Quand vous souhaitez revenir au mode d'affichage "portrait", il faut faire un nouveau saut de page et lui affecter le rôle **portrait** (*portrait* en anglais) :

```
//du contenu tout plein ici !  
  
[role=landscape]  
<<<  
  
// du contenu qui s'affiche au format paysage  
  
[role=portrait]  
<<<  
  
// le contenu s'affiche en mode portrait
```



Figure 103. rendu d'un document pdf avec une page en mode paysage

N'est-ce pas d'une simplicité redoutable ? (oui, je sais, pour Chuck, tout est simple...)

Il faut absolument faire des tests dans votre IDE pour assimiler ces notions.



Figure 104. un mème pour le plaisir

Source : <https://9gag.com/tag/chuck-norris>

# 19. Les attributs de document

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 21:10 | Auteur : Emmanuel Ravrat

Durée de réalisation : 60min

## 19.1. Un attribut, c'est quoi ?

Les **attributs de document** sont des noms associés à une valeur (du texte, un nombre) ou à "rien". Dans ce dernier cas, c'est l'existence ou l'inexistence de l'attribut qui est utile puisqu'il n'a pas de valeur.

Un attribut est une **variable**. Une variable peut être vue comme une boîte qui contient quelque chose. Sur cette boîte, il y a un nom qui permet de la différencier des autres boîtes. A partir du nom de la boîte, vous pouvez accéder au contenu.

Parfois, la boîte ne contiendra rien du tout, mais le fait que la boîte existe peut être utile.

L'existence des attributs est un énorme avantage sur les traitements de texte tels que Word, Writer, etc.

Imaginez une boîte qui a un nom. A chaque fois que vous utilisez son nom, vous avez accès à son contenu. Ne serait-ce pas génial si cela existait ?! Et bien, c'est ce que propose AsciiDoc avec ses attributs ! Cela permet d'utiliser le nom d'un attribut n'importe où dans le document afin d'accéder à son contenu !

Nous verrons plus loin comment profiter de cette incroyable fonctionnalité !

Il y a deux types d'attributs :

- les **attributs intégrés**

Le langage AsciiDoc contient déjà des attributs qui permettent de configurer son fonctionnement. Ces attributs sont des **attributs intégrés** car ils font déjà partie du langage AsciiDoc.

Voici quelques usages des attributs intégrés :

- Donner accès aux informations du document
- Définir les métadonnées du document
- Activer ou désactiver les fonctionnalités intégrées
- Configurer les fonctionnalités intégrées
- Déclarer l'emplacement des ressources, comme les images

Source : <https://docs.asciidoc.org/asciidoc/latest/attributes/document-attributes/>

- les **attributs définis par l'utilisateur**

Lorsqu'un attribut est créé par l'utilisateur (celui qui rédige le contenu AsciiDoc), il s'agit alors d'un [attribut de l'utilisateur](#).



Un attribut ne doit pas être déclaré à l'intérieur d'un bloc délimité tel un bloc de code ([chapitre 15, Les blocs de code source](#)) ou encore un bloc d'avertissement ([chapitre 14, Les blocs d'avertissement \(admonitions\)](#)), un bloc littéral, etc.). Le comportement est "indéfini", c'est-à-dire qu'il n'est pas celui attendu mais sans savoir lequel.

## 19.2. Manipuler un attribut

### 19.2.1. Déclarer / créer un attribut

Pour créer un attribut, il faut écrire son nom et **l'encadrer par un double-point** :

```
//déclaration d'un attribut  
:nom_du_heros: ①
```

① l'attribut `nom_du_heros` est déclaré mais il n'a aucune valeur. Le nom d'attribut est encadré par le caractère `:`.

Dès lors qu'un attribut est déclaré, il est dit "activé" ou défini.



Dès lors qu'un attribut est déclaré, il est défini pour tout le reste du document, y compris pour les sous fichiers qui seraient inclus (pour l'inclusion de fichiers, voir le chapitre [chapitre 25, Les bonnes pratiques de création d'un document AsciiDoc](#) et plus précisément la [partie 25.4, "Partager un fichier pour ne pas se répéter"](#)).



Le nom de l'attribut doit commencer par une lettre, un chiffre ou un underscore `_`. Il ne doit pas contenir d'espace ni de point.

Il doit être créé sur une ligne vide et en dehors de tout bloc (d'avertissement, de code, etc.).



Asciidoc ne différencie pas les majuscules des minuscules. Il est fortement conseillé de n'utiliser que des minuscules dans le nom d'un attribut



J'utilise personnellement l'underscore pour séparer les mots qui composent le nom d'un attribut (cette pratique est considérée comme non valide par la documentation) sans jamais avoir rencontré de problème jusqu'à ce jour. Je trouve cela plus lisible et évite un conflit de nom avec un attribut intégré.



Evitez les caractères accentués dans les noms de vos attributs. Cela rend plus difficile

leur manipulation, notamment si plusieurs personnes doivent les manipuler. Par ailleurs, cela peut générer des problèmes avec le processeur AsciiDoc.

### 19.2.2. Affecter une valeur à un attribut

Pour affecter une valeur à un attribut, il faut écrire la valeur à la suite du nom en veillant à laisser au moins un espace après le dernier caractère `:` :

```
//affectation d'une valeur à un attribut  
:nom_du_heros: Chuck Norris ①
```

- ① l'attribut `nom_du_heros` est déclaré et associée à la valeur "Chuck Norris". Il y a au moins un espace entre la déclaration de l'attribut et sa valeur.

Il est possible d'écrire la valeur sur plusieurs lignes en utilisant le **caractère de continuation de ligne** `\`. Cela permet de faciliter la lecture lorsque le contenu textuel est assez long :

```
//affectation d'une valeur répartie sur plusieurs lignes (mais pas rendu lors de  
l'affichage)  
:heros: Chuck Norris, le plus grand, le plus beau, \ ①  
le plus fort, le plus mystique, \ ②  
le plus éternel ③
```

- ① première partie de la valeur affectée. Le caractère `\` indique à AsciiDoc que la ligne n'est pas terminée

- ② suite de la ligne précédente. Le caractère `\` indique à AsciiDoc que la ligne n'est pas terminée

- ③ suite de la ligne précédente. La ligne étant terminée, il n'y a plus de caractère `\`.

Si l'objectif est de forcer le retour à la ligne dans la valeur, il faut utiliser le caractère `+` :

```
//déclaration d'un attribut avec des retours à la ligne forcé (lors de l'affichage)  
:heros: Chuck Norris, le plus grand, le plus beau. + \ ①  
C'est également le plus fort, le plus mystique. + \  
Il est éternel.
```

- ① le caractère `+` permet de forcer le retour à la ligne lorsque la valeur de l'attribut sera affichée. Le caractère `\` indique que la valeur affectée à l'attribut continue sur la ligne suivante.

La valeur de l'attribut peut contenir du texte formaté. Ainsi, il est possible d'affecter la valeur suivante :

```
//déclaration d'un attribut dont la valeur contient du texte formaté  
:heros: Le grand *Chuck Norris* _(faut-il encore le rappeler ?)_
```

La valeur d'un attribut peut contenir la valeur d'un autre attribut. Dans ce cas, le nom de l'attribut

utilisé est encadré par des accolades :

```
//déclaration d'un premier attribut contenant le nom et prénom  
:nom_du_heros: Chuck Norris  
//déclaration d'un second attribut faisant appel à la valeur de l'attribut précédent  
:heros: Le grand *{nom_du_heros}* _(faut-il encore le rappeler ?)_
```

### 19.2.3. Afficher la valeur d'un attribut

Pour afficher la valeur d'un attribut, il faut simplement écrire le nom de l'attribut souhaité et l'encadrer d'accolades :

```
//déclaration et affectation d'un attribut  
:nom_du_heros: Chuck Norris  
  
//utilisation de l'attribut  
Le héros est sans surprise {nom_du_heros} ①
```

① l'attribut va être substitué (remplacé) par sa valeur

Tout cela fonctionne très bien mais si la valeur contient du texte formaté et / ou un autre attribut, le formatage n'est pas rendu et l'attribut non substitué :

```
//déclaration d'un premier attribut  
:nom_du_heros: Chuck Norris  
  
//déclaration d'un second attribut contenant un autre attribut et des caractères de  
formatage du texte.  
:heros: Le grand *{nom_du_heros}* _(faut-il encore le rappeler ?)_ ①  
  
//affichage du héros  
{heros} ②
```

① la valeur contient du texte formaté

② affichage de la valeur de l'attribut :

```
Le grand *Chuck Norris* _(faut-il encore le rappeler ?)_
```

Le formatage n'a pas été appliqué et l'attribut contenu dans la valeur n'a pas été remplacé par sa valeur !

Pour que le formatage contenu dans la valeur d'un attribut soit appliqué, il faut passer l'attribut à la **macro de passage en ligne** avec les arguments **a** et **q** :

```
//affichage du héros avec substitution de l'attribut contenu dans la valeur et  
application du formatage
```

```
pass:a,q[{\heros}] ①
```

① la macro en ligne `pass` indique qu'il faut faire un traitement du contenu passé entre crochet. Le caractère `a` pour `asciidoc` indique que si des attributs sont présents, ils doivent être substitués par leur valeur. Le caractère `q` indique que s'il y a des caractères de formatage, il faut les appliquer.

Rendu :

```
Le grand Chuck Norris (faut-il encore le rappeler ?)
```

Les attributs peuvent utilement être mobilisés dans un bloc littéral (un bloc de code par exemple). Attention, je ne parle pas de **créer** un attribut dans un bloc de code, cela ne doit jamais être fait. Je parle d'afficher la valeur d'un attribut dans un bloc de code.

Imaginons que vous souhaitez documenter l'installation de php sur votre machine Windows via [Chocolatey](#). Vous allez écrire le contenu AsciiDoc ci-dessous dans votre documentation :

```
[source,powershell]
-----
choco install php --version=8.1.0
-----
```

Le problème est que la version de PHP va évoluer et il faudra alors mettre la documentation à jour partout où vous avez mentionné la version `8.1.0`. Cela conduit à des oubli et/ou à des erreurs.

Pour éviter cela, vous pouvez stocker le numéro de version dans un attribut puis utiliser ce dernier dans le bloc de code :

```
//affectation du numéro de version dans un attribut
:version_php: 8.1.0

[source,powershell]
-----
choco install php --version={version_php} ①
-----
```

① l'attribut `version_php` contient la version de php.

Cependant, le rendu n'est pas celui attendu car l'attribut n'est pas substituée par sa valeur :

```
choco install php --version={version_php}
```

Pour que l'attribut soit substitué, il faut le demander à AsciiDoc. Cela passe par l'attribut `subs` et la valeur `attributes+` (avec un `+` à la fin) :

```
//affectation du numéro de version dans un attribut
:version_php: 8.1.0

[source,powershell, subs=attributes+] ①
-----
choco install php --version={version_php} ②
-----
```

① l'attribut `subs` indique à AsciiDoc qu'il doit faire des substitutions. La valeur `attribute` indique qu'il faut substituer les attributs. Le caractère `+` indique qu'il doit faire la substitution des attributs après avoir traité le contenu du bloc (pour faire simple, cela signifie "fait ce que tu dois faire pour rendre le contenu et quand tu as fini, tu remplaces les attributs par leur valeur")

② l'attribut sera bien substitué par sa valeur

Le rendu est bien celui attendu, l'attribut a été remplacé par sa valeur :

```
choco install php --version=8.1.0
```

Pour des besoins plus poussés, vous pouvez lire la [page de documentation dédiée à la substitution des attributs](#).

## 19.3. Les attributs intégrés

Les **attributs intégrés** sont des attributs qui existent déjà dans le langage AsciiDoc. C'est pour cela qu'ils sont dit "intégrés" puisqu'ils sont intégrés nativement dans le langage.



Un attribut intégré est réservé soit pour configurer le comportement d'AsciiDoc, soit pour ajouter des métadonnées.

Il existe une page de la documentation qui en fait la [liste complète](#).

Voici les attributs intégrés que je juge les plus utiles :

attribut intégré	valeurs pertinentes (pour nous)	utilité
<code>author</code>	nom de l'auteur	permet de préciser l'auteur du document
<code>chapter-signifier</code>	aucune valeur	étiquette ajoutée avant chaque titre de niveau 1. Ne mettre aucune valeur permet d'écraser la valeur par défaut qui est "Chapter" avec une chaîne vide.

attribut intégré	valeurs pertinentes (pour nous)	utilité
doctype	book / article	permet d'indiquer le type de document de sortie. Puisque notre objectif est de générer des fichiers pdf, nous n'utiliserons que le type <b>book</b> ou <b>article</b> (voir le chapitre <a href="#">chapitre 22, Le type de document</a> ). Ce type permet de gérer une hiérarchie de titres, les préfaces, un index, etc.
experimental		La seule déclaration de cet attribut permet d'activer les macros qui permettent de générer des boutons, des menus et des touches de clavier (pas de valeur attendue)
icons	font	indique que les icons doivent être utilisés pour les blocs d'avertissement.
imagesdir	une url de répertoire	dossier qui va contenir les images de votre document AsciiDoc. Je conseille très fortement de laisser la valeur vide et de suivre les indications du chapitre <a href="#">chapitre 10, Les images</a> .
revnumber	un numéro	<b>numéro de version</b> ou de révision du document. C'est très utile pour pouvoir distinguer les différentes versions d'un même document.
revdate	une date	date de version ou de révision du document. C'est un très bon complément à l'attribut <b>revnumber</b> .
sectnums		si déclaré, active la numérotation des titres (pas de valeur attendue).
sectnumlevels	0-5	Indique le niveau de titre maximum qui sera automatiquement numéroté.
toc		La seule déclaration de cet attribut permet d'activer l'affichage de la table des matières (pas de valeur attendue)
toclevels	1-5	Niveau de titre à afficher dans la table des matières
toc-title	Table des matières	Titre de la table des matières. Par défaut, le titre est "Table of content"



Tous ces attributs sont à déclarer sous le titre principal du document (voir [chapitre 5, Les titres](#)).

Voici un exemple de document AsciiDoc mobilisant plusieurs attributs intégrés. Ceux-ci sont déclarés sous le titre principal du document.

```
//titre principal du document (il ne doit y avoir aucun contenu à afficher avant ce titre)
= Titre principal du document
//le document généré doit être de type "livre"
:doctype: book
//les symboles utilisés dans les blocs d'avertissement sont des icônes issues de la police utilisée
:icons: font
//activation de la numérotation des titres
:sectnums:
//niveau de titre maximum à numéroter
:sectnumlevels: 5
//étiquette placée avant chaque chapitre (ici une chaîne vide)
:chapter-signifier:
//activation des macros de touche de clavier et de menu
:experimental:
//activation de l'affichage de la table des matières
:toc:
//niveau de titre maximum à faire figurer dans la table des matières
:toclevels: 5
//titre de la table des matières
:toc-title: Table des matières

//il faut laisser une ligne vide avant d'écrire le contenu du document.
== Premier chapitre

// ... ici du contenu
```

## 19.4. Les attributs de l'utilisateur

### 19.4.1. Qu'est-ce qu'un attribut de l'utilisateur ?

Un **attributs de l'utilisateur** est un **attribut de document**, c'est-à-dire une variable créée par l'utilisateur et utilisable dans tout le document.

Un attribut de l'utilisateur peut être déclaré n'importe où dans le document.

Les attributs de l'utilisateur sont très utiles :

- pour **ne pas avoir à répéter un même contenu** en l'écrivant à chaque fois
- pour conditionner le rendu d'un contenu



Je conseille de déclarer et d'affecter les attributs de l'utilisateur en début de fichier AsciiDoc pour ne pas avoir à parcourir le fichier pour les retrouver.

### 19.4.2. Des attributs de l'utilisateur pour ne pas se répéter

Imaginons que vous écrivez un document concernant un produit ayant la référence RX74AAA. Vous mentionnez la référence de nombreuses fois dans le document. Cependant, la référence vient à changer. Il faut alors changer la référence partout où elle est utilisée. Si elle est utilisée dans différents documents, la mise à jour devient plus difficile. Dans quels documents la référence a-t-elle été utilisée ?

Ce problème peut être évité en utilisant un attribut créé par l'utilisateur puis en l'utilisant par la suite dans le document.

```
//la référence du produit  
:reference_produit: RX74AAA
```

```
//du contenu AsciiDoc...  
//encore du contenu AsciiDoc
```

Le produit \*{reference\_produit}\* est le seul qui est approuvé par Chuck Norris.

//Du contenu AsciiDoc encore et encore

Chuck le dit lui-même : "Le produit {reference\_produit} est le seul qui me donne entière satisfaction".

Rendu :

Le produit **RX74AAA** est le seul qui est approuvé par Chuck Norris.

Chuck le dit lui-même : "Le produit RX74AAA est le seul qui me donne entière satisfaction".

Le changement de la valeur affectée à l'attribut suffit pour mettre à jour le contenu du document sans effort supplémentaire.



L'attribut doit être préalablement déclaré avant d'être affiché.

### 19.4.3. Des attributs de l'utilisateur pour conditionner le contenu à rendre

Le contenu à rendre dans le document généré à partir de la source AsciiDoc peut dépendre d'un contexte.

L'objectif est de choisir ce qui doit être rendu dans le fichier pdf final en fonction de la valeur d'un attribut. C'est très pratique pour personnaliser le contenu d'un document en fonction d'un contexte.

AsciiDoc permet de tester la valeur d'un attribut et en fonction du résultat du test, il est possible de choisir les lignes à afficher et celles qu'il ne faut pas afficher. Cela passe par l'utilisation de directives qui sont abordées dans le chapitre [chapitre 24, Afficher / masquer du texte avec des conditions](#). Cependant, même si vous ne connaissez pas encore ces directives, cela n'empêche pas d'illustrer cet usage.

### Voici un premier exemple de cet usage :

Imaginons que vous ayez une documentation à faire pour des utilisateurs Windows et la même documentation pour des utilisateurs Mac. Par exemple, vous rédigez une documentation qui indique comment mettre à jour Python sur ces systèmes.

Vous pouvez l'indiquer en précisant pour chaque système la commande à jouer :

Mise à jour de Python sous Mac :

```
brew update && brew upgrade python
```

Mise à jour de Python sous Windows :

```
choco upgrade python
```

L'utilisateur Mac voit la commande Windows alors que cela n'a aucun intérêt.

L'utilisation d'un attribut de l'utilisateur qui stocke le système d'exploitation utilisé permet de choisir la commande à rendre dans le document final.

Voici comment faire cela :

Commande pour mettre à jour de Python :

```
//attribut contenant le nom de l'os
:os_name: mac

//si la valeur de l'attribut est "mac", alors on affiche la commande
ifeval::["{os_name}" == "mac"]
[source,console]
-----
brew update && brew upgrade python
-----
endif::[]

//si la valeur de l'attribut est "windows", alors on affiche la commande
ifeval::["{os_name}" == "windows"]
[source,powershell]
-----
choco upgrade python
```

```
----  
endif::[]
```

Rendu :

```
brew update && brew upgrade python
```

Cela signifie qu'il est possible de générer une version pdf pour les utilisateurs Mac et une autre version pour les utilisateurs Windows. Il suffit seulement de modifier la valeur avant de générer le pdf.

### **Voici un second exemple de cet usage :**

Vous voulez générer un support qui contient des questions et des réponses. Cependant, vous voulez pouvoir distribuer une version sans les réponses et une version avec les réponses.

Cela revient à prévoir un attribut qui lorsqu'il contient la valeur 1 indique qu'il faut afficher les réponses.

Voici ce que cela donne :

```
//état de l'affichage de la correction  
:afficher_la_correction: 0  
  
//une question  
Quelle est la force gravitationnelle de Chuck Norris ?  
  
ifeval::[{afficher_la_correction}==1]  
*Réponse :*  
  
La force gravitationnelle de Chuck Norris est si puissante qu'elle ne se conforme pas  
aux lois de la physique telles que nous les connaissons !  
  
Cette réponse ne peut être traitée que par Chuck lui-même !  
endif::[]
```

Rendu avec la valeur 0 :

```
Quelle est la force gravitationnelle de Chuck Norris ?
```

Rendu avec la valeur 1 :

```
Quelle est la force gravitationnelle de Chuck Norris ?
```

### Réponse :

La force gravitationnelle de Chuck Norris est si puissante qu'elle ne se conforme pas aux lois de la physique telles que nous les connaissons !

Cette réponse ne peut être traitée que par Chuck lui-même !

Si vous avez 30 questions, chaque réponse doit être gérée de la même façon. Ainsi, il est facile de générer un support sans les réponses et un support avec les réponses. Cela peut paraître fastidieux à écrire, mais avec les lives templates, cela ne demande plus aucun effort. (voir le chapitre sur les lives templates).

## 19.5. Désactiver un attribut

Il est parfois nécessaire de désactiver un attribut.

Désactiver un attribut revient à demander à AsciiDoc de considérer que l'attribut désactivé n'existe pas.

Pour désactiver un attribut, il suffit de préfixer le nom d'attribut par un point d'exclamation.

Voici un exemple :

```
//déclaration et affectation de l'attribut  
:age_de_chuck: 83000 chuckpunch  
  
//l'attribut est substitué par sa valeur  
Chuck Norris a {age_de_chuck} (_avec l'attribut actif_)  
  
//désactivation de l'attribut.  
:!age_de_chuck: 83000 chuckpunch ①  
  
//l'attribut n'existe plus, AsciiDoc ne peut le substituer par une valeur  
Chuck Norris a {age_de_chuck} (_avec l'attribut désactivé_)
```

① le point d'exclamation désactive l'attribut qui le suit. C'est comme s'il n'existant pas.

Rendu :

Chuck Norris a 83000 chuckpunch (*avec l'attribut actif*)

Chuck Norris a {age\_de\_chuck} (*avec l'attribut désactivé*)

Si vous voulez aller plus loin concernant les attributs, voici quelques liens de la documentation officielle :

- [les attributs de document](#)

- la substitution des attributs
- déclarer des attributs sur une ligne non vide, par exemple dans une cellule de tableau.
- la macro de passage en ligne `pass:[]`
- gérer les références vers des attributs désactivés ou inexistant

# 20. Les compteurs

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 21:11 | Auteur : Emmanuel Ravrat

Durée de réalisation : 20min

Les **compteurs** sont utiles pour créer une séquence. Il s'agit tout simplement de profiter de l'incrémentation automatique d'une valeur d'attribut à chaque fois que cet attribut est préfixé par `counter`.



Les compteurs ne doivent pas être utilisés en dehors du besoin simple d'afficher une valeur incrémentée.

Pour bien comprendre les compteurs, il faut avoir compris le chapitre sur les [attributs de l'utilisateur](#).

Une valeur d'attribut est incrémentée dès lors que l'attribut qui la contient est préfixé par `counter` :

```
//affectation de la valeur de départ de l'attribut qui sert de compteur  
:numero_de_question: 0 ①  
  
Question {counter:numero_de_question} : ②  
ceci est la question {numero_de_question} ③  
  
Question {counter:numero_de_question} : ②  
ceci est la question {numero_de_question} ③
```

- ① initialisation de la valeur du compteur à 0 (ce pourrait être une autre valeur). Lorsqu'un attribut est utilisé comme compteur sans valeur de début définie par l'utilisateur, la première valeur est 1. La valeur de compteur doit être une valeur qu'il est possible d'incrémenter.
- ② l'attribut `counter` placé juste avant l'attribut va incrémenter la valeur de ce dernier **puis** la substitution est réalisée.
- ③ affichage de la valeur courante de l'attribut

Voici le rendu du code AsciiDoc précédent :

```
Question 1 : ceci est la question 1
```

```
Question 2 : ceci est la question 2
```

Il peut être utilisé une lettre à la place d'une valeur numérique.

```
:lettre_question: A ①
```

```
Question {counter:lettre_question} :  
ceci est la question {lettre_question}
```

```
Question {counter:lettre_question} :  
ceci est la question {lettre_question}
```

① Le numéro de question est initialisé avec une lettre

Voici le rendu du code AsciiDoc précédent :

```
Question B : ceci est la question B
```

```
Question C : ceci est la question C
```

Comme le numéro de question a été initialisé à A, la première valeur affichée est B.

Cela peut être évité soit en n'incrémentant pas la première question, soit en utilisant le compteur lors de son premier appel :

```
//l'attribut compteur est initialisé à sa première utilisation  
Question {counter:lettre_question:A} : ①  
ceci est la question {lettre_question}  
  
Question {counter:lettre_question} :  
ceci est la question {lettre_question}
```

① la valeur du compteur est initialisée à A et A sera affiché.

Rendu obtenu :

```
Question A : ceci est la question A
```

```
Question B : ceci est la question B
```

# 21. Les annexes d'un document

Version 1 | Dernière mise à jour : 25/07/2024 à 17:35 | Auteur : Emmanuel Ravrat

Durée de réalisation : 25min

## 21.1. Créer des parties pour les annexes

La partie **annexe** d'un document permet de compléter ce dernier par des graphiques, des données, des tableaux, des détails techniques, des illustrations, trop volumineux pour être inclus dans le corps principal du texte.

Ces annexes (*appendix* en anglais) servent à approfondir certains points sans alourdir le contenu principal.

Généralement, la partie des annexes est placée en fin de document, mais avant l'index (voir [chapitre 23, Générer un index](#)).

Pour créer une partie pour les annexes, il suffit d'écrire le titre qui est un sous-titre du titre principal et d'ajouter sur ce titre le rôle de bloc **appendix** :

```
= La véritable histoire de Chuck
:doctype: article ①

== chapitre 1

// du contenu ...

== chapitre 2

// du contenu ...

[appendix] ②
== Les mentras de Chuck Norris à dire aux WC ③

[appendix] ②
== Les exploits de Chuck Norris convertis en mèmes ③

[appendix] ②
== Raccourcis clavier de la pensée de Chuck Norris ③
```

① Le document est de type article (doctype **article**) (voir le [chapitre sur les types de document](#))

② rôle de bloc permettant d'identifier la partie comme étant une partie annexe.

③ **le niveau de titre doit être un titre de niveau de niveau 1 (deux signes ==) dans le cas de l'utilisation d'un doctype **article****

Rendu :

**Appendix A: Les mentras de Chuck Norris à dire aux WC**

**Appendix B: Les exploits de Chuck Norris convertis en mèmes**

**Appendix C: Raccourcis clavier de la pensée de Chuck Norris**

Figure 105. rendu pdf des titres des parties annexes

Vous pouvez constater que AsciiDoc ajoute une lettre incrémentée automatiquement pour facilement repérer les annexes. De plus, l'étiquette `Appendix` qui signifie "annexe" précède ce numéro.

Pour avoir un rendu qui montre comment sont gérées les annexes dans la table des matières avec AsciiDoc, il faut l'afficher et numérotter les chapitres.

```
= La véritable histoire de Chuck
:toc: ①
:sectnums: ②

//... suite du contenu
```

① activation de la table des matières

② affichage des numéros de chapitres

## Table of Contents

1. chapitre 1 .....	1
2. chapitre 2 .....	1
Appendix A: Les mentras de Chuck Norris à dire aux WC .....	1
Appendix B: Les exploits de Chuck Norris convertis en mèmes .....	1
Appendix C: Raccourcis clavier de la pensée de Chuck Norris .....	1

Figure 106. gestion des annexes par AsciiDoc dans la table des matières

Les annexes ne sont pas considérées comme des chapitres par AsciiDoc ce qui est tout à fait normal.

Si le type de document était un livre (type `book`), le niveau de titre à utiliser pour les titres des annexes est le niveau 0 :



```
= La véritable histoire de Chuck
:doctype: book ①

== chapitre 1

// du contenu ...
```

```
== chaptire 2

// du contenu ...

[appendix]
= Les mentras de Chuck Norris à dire aux WC ②

[appendix]
= Les exploits de Chuck Norris convertis en mèmes ②

[appendix]
= Raccourcis clavier de la pensée de Chuck Norris ②
```

① Le type de document est fixé sur `book`. (voir le [chapitre sur les types de document](#))

② Le niveau de titre d'une partie d'annexe doit être le niveau 0 lorsque le doctype est `book`.



Pour la suite du cours, je considère que nous utilisons le [type de document article](#).

Une partie d'annexe peut contenir des sous-parties. Dans ce cas, les sous-parties ne doivent pas être associées au rôle de bloc `appendix` :

```
= La véritable histoire de Chuck
:doctype: article // c'est le type par défaut lorsque rien n'est précisé
:sectnums:
:toc:

== chapitre 1

== chaptire 2

[appendix]
== Les mentras de Chuck Norris à dire aux WC

==== Mentrás de la poussée obscure ①

==== Mentrás de l'après ①

[appendix]
== Les exploits de Chuck Norris convertis en mèmes

[appendix]
== Raccourcis clavier de la pensée de Chuck Norris
```

① sous-partie d'une partie d'annexe. Il ne faut pas appliquer le rôle `appendix` sur le titre.

# Table of Contents

1. chapitre 1 .....	1
2. chapitre 2 .....	1
Appendix A: Les mentras de Chuck Norris à dire aux WC .....	1
A.1. Mentrals de la poussée obscure .....	1
A.2. Mentrals de l'après .....	1
Appendix B: Les exploits de Chuck Norris convertis en mèmes .....	1
Appendix C: Raccourcis clavier de la pensée de Chuck Norris .....	1

Figure 107. rendu dans la table des matières avec les sous-parties

Les sous-parties de la partie A ont été automatiquement numérotées. C'est vraiment très pratique, car il n'y a jamais à se soucier de gérer la numérotation des annexes.

**Vous pouvez créer des liens vers les annexes comme vous le faites avec des titres :**

```
1 [[annexe_exploits_chuck_norris_convertis_en_memes]] ①
2 [appendix]
3 == Les exploits de Chuck Norris convertis en mèmes
```

① nom d'ancre qui doit être unique et qui va être utilisée dans un lien.

Ensuite, depuis un autre endroit dans le document :



```
1 //lien sans texte personnalisé. Le texte du lien sera automatiquement
celui du titre de l'annexe.
2 <<annexe_exploits_chuck_norris_convertis_en_memes>> ①
3
4 //lien avec texte personnalisé. Le texte du lien est celui qui est
précisé.
5 <<annexe_exploits_chuck_norris_convertis_en_memes, voir l'annexe des
exploits>> ①
```

① lien pointant l'ancre.

## 21.2. Personnaliser l'étiquette des annexes

Nous avons vu que AsciiDoc ajoute le terme "Appendix" avant le titre de chaque partie d'annexes.

Ce comportement est personnalisable grâce à l'attribut de document `appendix-caption` à définir sous le titre principal du document :

```
= La véritable histoire de Chuck
:toc:
:sectnums:
```

```
:appendix-caption: Annexe
```

```
//...
```

## Table of Contents

1. chapitre 1 .....
2. chaptire 2 .....
Annexe A: Les mentras de Chuck Norris à dire aux amis .....
A.1. Mentrás de la poussée obscure .....
A.2. Mentrás de l'après .....
Annexe B: Les exploits de Chuck Norris convertis en vers .....
Annexe C: Raccourcis clavier de la pensée de Chuck .....

Figure 108. personnalisation de l'étiquette des annexes

# 22. Le type de document

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 22:45 | Auteur : Emmanuel Ravrat

Durée de réalisation : 15min

## 22.1. Choisir le type de document de sortie

Le **type de document** ou **doctype** permet de déterminer la structure du document en sortie. Il existe les types **article**, **book**, **manpage** et **inline**.

Je n'aborderai que les types **article** et **book**.

### Le type **article**

Lorsque vous ne précisez rien, le type de document est **article**. Cela répond aux besoins les plus courants (comptes rendus, devoirs, documentations, etc.). Ce type de document permet d'inclure des **annexes**, un **index** entre autres choses.

En principe, c'est ce type de document que vous devriez utiliser pour des documents qui n'ont pas à être organisés en chapitres.

### Le type **book**

Ce type reprend les caractéristiques du type **article** mais permet en plus d'ajouter une page de garde, une bibliographie, un colophon, une préface, une dédicace.

Le rendu pdf obtenu avec de type de document diffère de celui obtenu avec le type **article**.

Avec le type **book**, votre document aura entre autres les caractéristiques suivantes :

- un **saut de page forcé entre chaque chapitre** (notion qui n'a pas de sens dans un article). Cela signifie qu'un nouveau chapitre commence sur une nouvelle page. Le document que vous êtes en train de lire utilise le doctype **book**.
- une **page de garde**
- gestion d'une **bibliographie** (non couvert par ce support de formation)

Vous devez utiliser ce type de document dès lors que vous souhaitez écrire des supports organisés en chapitres, que vous avez besoin d'une page de garde (**chapitre 29, La page de garde**), d'une bibliographie, etc.)

Si vous voulez en savoir un peu plus sur les types de document, vous pouvez lire cette **page de la documentation**

## 22.2. Spécifier le type de document

Le type de document est précisé avec l'attribut **doctype** (voir **chapitre 22, Le type de document**)

= Titre principal du document

```
:doctype: book ①
```

① ici, le type de document est un livre. Si rien n'est précisé, c'est équivalent à la déclaration suivante : `:doctype: article`. Il est préférable de spécifier le type de façon explicite même s'il s'agit d'un article.

- Créez un document avec un titre principal et quelques sous-titres de niveau 1.
- Ne spécifiez aucun doctype et générez le fichier pdf.
- Recommencez en précisant le doctype avec la valeur `book`.  
Vous devriez voir que le document contient une page de garde et que chaque chapitre débute sur une nouvelle page.
- Modifiez le doctype pour `article` puis regénérez le pdf.  
Vous n'avez plus de page de garde et les chapitres sont de simples parties qui se suivent sur la même page.

# 23. Générer un index

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 22:50 | Auteur : Emmanuel Ravrat

Durée de réalisation : 20min

## 23.1. Qu'est-ce qu'un index ?

Un **index** liste des mots clés qui apparaissent dans le document ou qui sont associés à un emplacement dans le document. A partir d'un mot clé situé dans l'index, il est possible de naviguer vers son emplacement dans le document.

### Index

#### A

- alignement horizontal, 81
- alterner entre ligne ombrée et non ombrée, 96
- ancre de référence croisée, 46
- attribut, 22
  - affectation, 102
  - afficher la valeur, 103
  - création, 102
  - désactiver, 111
- attribut de document, 23
- attributs de document, 23, 101
- attributs de l'utilisateur, 108
- attributs intégrés, 101, 106

#### B

- bloc de liste AsciiDoc, 58

- lien relatif vers un fichier, 44
- ligne d'un tableau, 74
- liste avec une image, 34
- liste imbriquée, 33
- liste non ordonnée, 31, 32
- liste non ordonnée imbriquée dans une ordonnée, 37
- Liste numérotée, 35
- liste ordonnée, 34
- liste ordonnée imbriquée dans une ordonnée, 36
- Liste ordonnées avec des lettres, M

#### M

- macro de passage en ligne, 104
- multiplicateur de colonne, 76

Figure 109. extrait d'un index pour illustration

## 23.2. Afficher l'index

Je commence par vous indiquer comment afficher l'index et ensuite je vous explique comment ajouter des entrées dans celui-ci.

Un index se place à la fin du document même s'il est techniquement possible de le placer ailleurs avec AsciiDoc.

L'ajout de l'index se fait par l'ajout du style de bloc **index** placé sur un titre de niveau 2 :

```
= Titre principal du document  
//...plein de contenu super intéressant...  
//c'est la fin du document, je veux ajouter un index
```

```
[index] ①  
== Index ②
```

① le style de bloc `index` indique à Asciidoc qu'il doit insérer toutes les entrées ajoutées à l'index sous le titre.

② titre de l'index. Le titre est totalement libre mais il doit être un sous-titre du titre principal.



L'index n'est rendu que dans le format de sortie pdf.

Si vous voulez que l'index débute sur une nouvelle page (ce qui n'est pas le cas par défaut), vous pouvez ajouter les caractères qui forcent le saut de page :



<<< ①

```
[index]  
== Index
```

① saut de page forcé

### 23.3. Ajouter des entrées à l'index

Pour ajouter un mot à l'index, il suffit d'entourer ce mot par des **doubles parenthèses**.

```
= La véritable histoire de Chuck  
// du contenu ...
```

```
//quelque part dans le document  
Chuck Norris est le ((maître ultime)) de tout ce qui est, a été et sera. ①
```

Cet état peut s'expliquer car il est à l'((origine de tout)). ②

```
//à la fin du document  
<<<  
[index]  
== Index
```

① la chaîne "maître ultime" est ajouté à l'index

② la chaîne "origine de tout" est ajouté à l'index

Rendu de l'index :



Figure 110. index généré automatiquement avec des entrées affichées dans le flux

Il est possible d'ajouter un terme à l'index sans que ce terme soit affiché dans le contenu. Dans ce cas, il faut utiliser une triple parenthèse :

//quelque part dans le document

Chuck Norris est le ((maître ultime)) de tout ce qui est, a été et sera. ①

Cet état peut s'expliquer car il est à l'((origine de tout)). ②

((force mentale)) ③

L'atout majeur de Chuck est capacité à faire plier l'existant par sa seule projection mentale.

Il pourrait ne jamais avoir à solliciter sa masse musculaire ((force physique)) mais le fait pour créer de la lumière divine. ④

① l'expression entre double parenthèse (( )) sera affichée dans le rendu.

② l'expression entre ((( ))) ne sera pas affichée dans le pdf ni dans la prévisualisation mais sera bien présent dans l'index. Le terme est associé au paragraphe qui lui succède.

③ l'expression entre ((( ))) ne sera pas affichée dans le pdf ni dans la prévisualisation mais sera bien présent dans l'index. Le terme est associé au mot qui le précède.

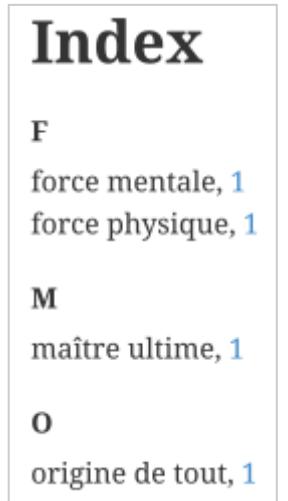


Figure 111. index généré automatiquement avec des entrées non affichées dans le flux

AsciiDoc offre une autre fonctionnalité intéressante concernant les entrées dans l'index. Il est possible de regrouper des termes à partir d'un mot clé "maître" et de lui ajouter des mots

complémentaires. Cela facilite la lecture de l'index et la recherche des termes est facilitée pour l'utilisateur.

Voici une illustration de cette fonctionnalité :

```
// Contenu à ajouter avant l'index
```

Chuck Norris utilise la force de son regard (((force, regard))) en complément de la force de la pousse de ses cheveux (((force, cheveux))) pour contrôler la force solaire (((force, solaire))) qui émane du soleil, c'est-à-dire de lui-même. ①

- ① Pour associer un mot à une entrée principale, il faut séparer le mot "maître" et le terme secondaire par une virgule.



Figure 112. entrées d'index associées à une même entrée dans l'index

Nous en avons terminé avec l'index.

# 24. Afficher / masquer du texte avec des conditions

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 22:59 | Auteur : Emmanuel Ravrat

Durée de réalisation : 40min

## 24.1. Conditionner l'affichage à l'existence d'un attribut



Cette partie nécessite d'avoir assimilé la notion d'attribut (voir [chapitre 19, Les attributs de document](#))

Avec AsciiDoc, il est possible de conditionner l'affichage de certaines lignes en fonction de l'existence d'un attribut.

Pour rappel, un attribut existe dès lors qu'il est déclaré même s'il n'a pas de valeur.

```
:mon_attribut: ①
```

① L'attribut `mon_attribut` est déclaré.

Dans AsciiDoc, un attribut qui est déclaré est dit "activé" ou "défini".

Imaginons que vous ayez un support avec des questions et des réponses. Vous souhaitez pouvoir générer un support sans les réponses et un support avec les réponses.

Vous pouvez alors conditionner l'affichage des réponses à l'existence d'un attribut. Cela revient à vérifier si l'attribut est **défini**.

AsciiDoc prévoit la directive `ifdef` qui signifie "si l'attribut est défini, alors fait ceci".

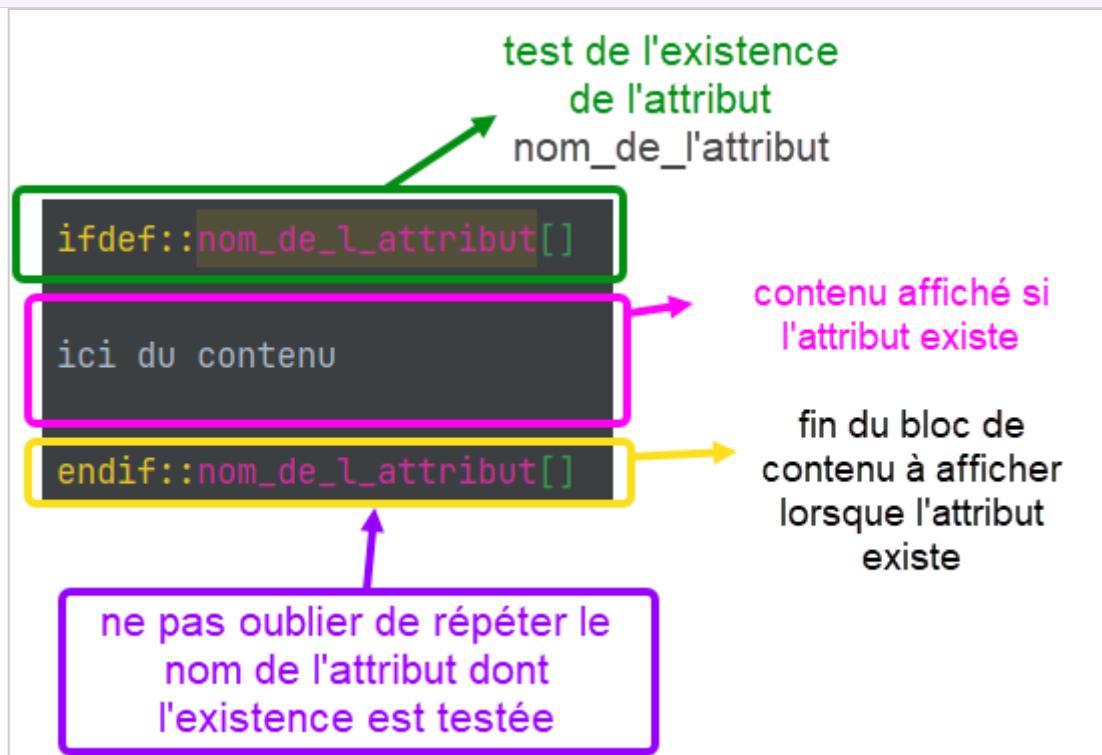


Figure 113. structure de test de la directive `ifdef`

Il n'est pas obligatoire de répéter le nom de l'attribut après `endif::` mais je le conseille fortement pour améliorer la lisibilité. Cela devient même indispensable en cas d'imbrication (utilisation de `ifdef` dans un `ifdef`)

 Pour ceux qui viennent du monde du développement, ils s'attendent peut être à trouver une structure alternative du type "si l'attribut est défini alors on affiche ce texte, sinon on affiche cet autre texte". Cela n'est (malheureusement) pas prévu par le langage AsciiDoc. La solution consiste à écrire un nouveau test pour gérer le cas alternatif.

Voici une question suivie d'une réponse dont l'affichage est conditionné à la définition de l'attribut `afficher_reponse` :

```
//question
Quelle est la couleur du cheval blanc de Chuck Norris ?

ifdef::afficher_reponse[] ①
*Réponse*

Chuck dit que son cheval est de couleur norris. ②
//et encore du contenu, n'importe lequel (liste, bloc, tableau, etc.)
endif::afficher_reponse[]
```

① le test vérifie que l'attribut `afficher_reponse` est défini ou déclaré.

② lignes affichées si le test est vrai, c'est-à-dire que l'attribut a été déclaré / est défini.

## Quelle est la couleur du cheval blanc de Chuck Norris ?

Figure 114. rendu du test sans que l'attribut `afficher_reponse` ait été défini

Maintenant, définissons l'attribut `afficher_reponse` en le déclarant.

```
:afficher_reponse: ①

//question
Quelle est la couleur du cheval blanc de Chuck Norris ?

ifdef::afficher_reponse[] ②
*Réponse*

Chuck dit que son cheval est de couleur norris. ③
//et encore du contenu, n'importe lequel (liste, bloc, tableau, etc.)
endif::afficher_reponse[]
```

① L'attribut `afficher_reponse` est déclaré (donc défini)

② le test qui évalue si l'attribut `afficher_reponse` est défini va donner "vrai"

③ le texte placé entre `ifdef` et `endif` est affiché puisque le test est vrai.

## Quelle est la couleur du cheval blanc de Chuck Norris ?

### Réponse

Chuck dit que son cheval est de couleur norris.

Figure 115. rendu du test sans que l'attribut `afficher_reponse` ait été défini

 Dès lors qu'un attribut est déclaré, il est défini pour tout le reste du document, y compris pour les sous fichiers qui seraient inclus (pour l'inclusion de fichiers, voir le chapitre [chapitre 25, Les bonnes pratiques de création d'un document AsciiDoc](#) et plus précisément la [partie 25.4, "Partager un fichier pour ne pas se répéter"](#).

 Il est possible d'imbriquer des directives `ifdef`.

Nous venons de voir la directive `ifdef` qui évalue si un attribut est défini. AsciiDoc permet également de tester si un attribut n'est pas défini.

Un **attribut indéfini** est un attribut qui n'existe pas, qui n'a jamais été déclaré ou qui a été désactivé.

Nous pouvons reprendre le test précédent en conditionnant l'affichage des réponses à l'inexistence de l'attribut `masquer_les_responses` :

```
:masquer_les_responses:
```

```
//question  
Pourquoi Chuck Norris ne joue-t-il jamais au jeu de la chaise musicale ?  
  
ifndef::masquer_les_reponses[] ①  
*Réponse*
```

Parce qu'à chaque fois qu'il s'assoit, il brise la chaise... et le sol... et crée un nouveau cratère sur Terre !  
endif::masquer\_les\_reponses[] ②

① si l'attribut `masquer_les_reponses` est indéfini (non déclaré), le contenu entre `ifndef` et `endif` est affiché.

② fin du bloc `ifndef`

Pourquoi Chuck Norris ne joue-t-il jamais au jeu de la chaise musicale ?

Figure 116. rendu du test alors que `masquer_les_reponses` est défini

La réponse n'est pas affichée puisque l'attribut `masquer_les_reponses` est défini. Le test vérifie que l'attribut **n'est pas défini**.

Si nous retirons l'attribut, il devient indéfini :

```
// :masquer_les_reponses: ①  
  
//question  
Pourquoi Chuck Norris ne joue-t-il jamais au jeu de la chaise musicale ?  
  
ifndef::masquer_les_reponses[] ②  
*Réponse*  
  
Parce qu'à chaque fois qu'il s'assoit, il brise la chaise... et le sol... et crée un nouveau cratère sur Terre !  
endif::masquer_les_reponses[]
```

① L'attribut est mis en commentaire ce qui revient à ne pas l'écrire. C'est pratique pour activer, désactiver rapidement un attribut.

Pourquoi Chuck Norris ne joue-t-il jamais au jeu de la chaise musicale ?

### Réponse

Parce qu'à chaque fois qu'il s'assoit, il brise la chaise... et le sol... et crée un nouveau cratère sur Terre !

Figure 117. rendu du test alors que `masquer_les_reponses` est indéfini

AsciiDoc prévoit la possibilité de désactiver un attribut. Cela est parfois nécessaire lorsqu'il faut changer un état après avoir déclaré / défini l'attribut (par exemple dans un sous fichier, plus loin dans le document, etc.) Le symbol bang ! permet d'inverser le sens d'une déclaration.

```
//déclaration de l'attribut
:masquer_les_reponses: ①

ifdef::masquer_les_reponses[] ②
Ce contenu sera affiché puisque l'attribut 'masquer_les_responses' est défini.
endif::masquer_les_reponses[]

ifndef::masquer_les_reponses[] ③
Ce contenu ne sera pas affiché puisque l'attribut 'masquer_les_responses' est défini.
endif::masquer_les_reponses[]

//l'attribut est désactivé
:!masquer_les_reponses: ④

ifdef::masquer_les_reponses[] ⑤
Ce contenu ne sera pas affiché puisque l'attribut 'masquer_les_responses' est désactivé avec l'opérateur '!'
endif::masquer_les_reponses[]

ifndef::masquer_les_reponses[] ⑥
Ce contenu sera affiché puisque l'attribut 'masquer_les_responses' est désactivé donc
indéfini avec l'opérateur '!'.
endif::masquer_les_reponses[]
```

- ① l'attribut est déclaré, il est donc défini pour tout le reste du document
- ② le test est "vrai" car l'attribut est défini. Le contenu du bloc sera affiché.
- ③ le test est "faux" car l'attribut n'est pas indéfini. Le contenu du bloc ne sera pas affiché.
- ④ l'attribut est désactivé grâce à l'opérateur "bang" !. Il n'existe plus à partir de cette ligne pour tout le reste du document (tant qu'il n'est pas déclaré de nouveau).
- ⑤ l'attribut étant désactivé, il est indéfini. Le test est "faux". Le contenu du bloc n'est pas affiché.
- ⑥ l'attribut étant désactivé, il est indéfini. Le test est "vrai". Le contenu du bloc sera affiché.

Ce contenu sera affiché puisque l'attribut `masquer_les_responses` est défini.

Ce contenu sera affiché puisque l'attribut `masquer_les_responses` est désactivé donc indéfini avec l'opérateur !.

*Figure 118. rendu avec activation et désactivation d'un attribut avec !*

L'écriture des tests est pénible mais en utilisant des lives template adaptés, cela devient très simple et très rapide !

Par exemple, j'écris `ifdef`, je tabule et je n'ai plus qu'à saisir le nom de l'attribut à tester et le contenu à afficher. (voir le chapitre sur les lives template)

La structure `ifdef` ou `ifndef` peut être écrite sur une ligne lorsque le contenu textuel à afficher est court. Dans ce cas d'utilisation, le texte est à placer entre les crochets.

ifdef::nom\_attribut[Cette phrase s'affiche si l'attribut est défini.] ①

### ① Utilisation de la structure `ifdef` sur une seule ligne

Pour en savoir plus sur les directives `ifdef` et `ifndef`, vous pouvez lire la [page de la documentation dédiée à ce sujet](#)

## 24.2. Conditionner l'affichage à la valeur d'un attribut

Dans la partie précédente, l'affichage de contenu a été conditionné à l'activation ou la désactivation d'un attribut, ou autrement dit au fait qu'il existe ou qu'il n'existe pas.

Cela est insuffisant lorsqu'il peut y avoir plus de 2 états. Effectivement, avec `ifdef` et `ifndef`, seuls 2 états peuvent être testés : l'attribut existe ou n'existe pas / est désactivé).

En affectant une valeur à un attribut, il devient possible d'avoir un nombre d'états théoriquement infini. Il suffit de comparer la valeur stockée dans l'attribut à une valeur définie.

AsciiDoc permet cela grâce à la directive `ifeval`.

Voici pour illustrer mon propos une documentation qui donne la commande permettant de mettre python à jour sur les systèmes d'exploitation Mac, Linux et Windows.

Voici la commande qui permet la mise à jour de python sur votre système :

Sous Mac :

```
.mise à jour de python sous Mac  
[source,console]  
----  
$ brew update && brew upgrade python  
----
```

Sous Windows :

```
.mise à jour de python sous Windows  
[source,console]  
----  
> choco upgrade python  
----
```

Sous Linux :

```
.mise à jour de python sous Linux  
[source,console]  
----  
$ sudo apt install python3.9  
----
```

L'idéal serait de générer trois documentations (une par système d'exploitation). L'utilisateur n'aurait ainsi que les commandes qui l'intéressent.

Cela peut être fait en créant un attribut qui va stocker le nom du système d'exploitation. Ensuite, il faut comparer la valeur de l'attribut au nom du système pour lequel afficher la commande :

```
:os_name: Windows ①
```

Voici la commande qui permet la mise à jour de python sur votre système \*{os\_name}\* :

```
②
```

```
ifeval::["{os_name}" == "MacOS"] ③
```

```
.mise à jour de python
```

```
[source,console]
```

```
----
```

```
$ brew update && brew upgrade python
```

```
----
```

```
endif::[]
```

```
ifeval::["{os_name}" == "Windows"] ④
```

```
.mise à jour de python
```

```
[source,console]
```

```
----
```

```
> choco upgrade python
```

```
----
```

```
endif::[]
```

```
ifeval::["{os_name}" == "Linux"] ④
```

```
.mise à jour de python
```

```
[source,console]
```

```
----
```

```
$ sudo apt install python3.9
```

```
----
```

```
endif::[]
```

① déclaration de l'attribut qui va contenir le nom du système d'exploitation et affectation d'une valeur

② la valeur de l'attribut est utilisée pour personnaliser la phrase qui annonce la commande

③ la directive `ifeval` vérifie que la valeur stockée dans `os_name` est égale à la chaîne de caractères `MacOs`. Si c'est le cas, le contenu placé entre `ifeval` et `endif` est affiché.

④ comparaison de la valeur stockée dans l'attribut `os_name` à la valeur située à droite de l'opérateur  
`==`

Voici la commande qui permet la mise à jour de python sur votre système **Windows** :

*mise à jour de python*

```
> choco upgrade python
```

Figure 119. rendu personnalisé en fonction de la valeur de l'attribut `os_name`

Lorsque vous comparez des chaînes de caractères, il faut les encadrer de guillemets " sans quoi l'expression évaluée donnera "faux".

L'erreur courante est d'écrire les tests suivants :

 //erreur 1 : la substitution de l'attribut n'est pas encadrée de guillemets  
ifeval:::{os\_name} == "Linux"  
  
//erreur 2 : la valeur de comparaison n'est pas encadrée de guillemets alors que c'est une chaîne de caractères  
ifeval:::"{os\_name}" == Linux  
  
//erreur 3 : utiliser un seul signe = pour faire la comparaison  
ifeval:::"{os\_name}" = "Linux"

S'il faut comparer une valeur numérique, il ne faut pas de guillemets autour des valeurs :

```
ifeval:::[30>10] ①
//...
endif::[]

:age: 40

ifeval:::{age} >= 18] ②
Voici un contenu susceptible de choquer les plus jeunes...
//ici du contenu sensible
endif::[]
```

① deux nombres sont comparés. Ils ne sont pas encadrés de guillemets

② l'attribut `age` est un entier, il peut être comparé à `18` qui est du même type. Ils ne sont pas encadrés de guillemets

 Contrairement à la directive `ifdef` ou `ifndef`, la fin de la structure `endif` ne contient pas d'attribut :

```
ifdef::os_name[]
//...
```

```
endif::os_name[] ①

ifeval::["{os_name}" == "MacOs"]
//...
endif::[] ②
```

- ① la macro `endif::` peut être suivi de l'attribut testé dans le cas d'une structure `ifdef` (ou `ifndef`).
- ② la macro `endif::` n'est suivi de rien dans le cas d'une structure `ifeval`.

Lorsque le contenu dans la structure `ifeval` contient lui-même d'autres structures `ifdef`, `ifndef`, `ifeval`, il peut être utile d'ajouter un commentaire à la fin de chaque structure pour mieux s'y retrouver.



```
1 ifeval::{age}>18]
2 ifeval::{niveau_competence} > 20]
3 //...
4 //end eval niveau_competence ①
5 endif::[]
6 //end eval age ②
7 endif::[]
```

- ① le commentaire permet de savoir que le `endif` de la ligne 5 est lié au `ifeval` de la ligne 2 grâce au rappel de l'attribut évalué
- ② le commentaire permet de savoir que le `endif` de la ligne 7 est lié au `ifeval` de la ligne 1 grâce au rappel de l'attribut évalué

Il ne faut pas comparer un attribut désactivé ou inexistant avec une valeur à moins de le faire en connaissance de cause :



```
ifeval::{attribut_inexistant} == 20] ①
Cette ligne ne sera pas affichée
endif::[]

ifeval::{attribut_inexistant} != 20] ②
Cette ligne est affichée
endif::[]
```

- ① L'attribut n'existe pas, il ne peut être égal à aucune valeur, même si c'est une chaîne vide.
- ② l'attribut n'existe pas, il est forcément différent de toute valeur.

Voici la liste des opérateurs mobilisables dans un test avec `ifeval` :

Table 9. tableau des opérateurs de comparaison

opérateur de comparaison	signification
<code>==</code>	Vérifie si les deux valeurs sont égales
<code>!=</code>	Vérifie si les deux valeurs sont différentes
<code>&lt;</code>	Vérifie que la valeur à gauche de l'opérateur est strictement plus petite que celle située à sa droite
<code>&lt;=</code>	Vérifie que la valeur à gauche de l'opérateur est plus petite ou égale que celle située à sa droite
<code>&gt;</code>	Vérifie que la valeur à gauche de l'opérateur est strictement plus grande que celle située à sa droite
<code>&gt;=</code>	Vérifie que la valeur à gauche de l'opérateur est plus grande ou égale que celle à sa droite



Les deux côtés comparés doivent avoir le même type de valeur. Si ce n'est pas le cas, la comparaison échouera. Si la comparaison échoue, la condition sera évaluée comme fausse.

Pour en savoir un peu plus sur la directive `ifeval`, vous pouvez lire  [cette page de la documentation](#)

Faites des tests afin de manipuler les attributs et leur valeur pour afficher ou masquer du texte. **C'est la pratique qui vous fera comprendre et assimiler ces notions.**

# 25. Les bonnes pratiques de création d'un document AsciiDoc

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 23:28 | Auteur : Emmanuel Ravrat

Durée de réalisation : 1h 30min

Ce chapitre est très important et doit absolument être réalisé. Une simple lecture ne suffit pas (comme pour tous les autres chapitres, mais celui-ci encore plus !).

## 25.1. Un fichier AsciiDoc ne doit pas être "long"

### 25.1.1. Les inconvénients d'un fichier trop "long"

Travailler avec un long fichier AsciiDoc peut sembler pratique au premier abord car **tout est centralisé au même endroit**.

Cependant, lorsque le fichier devient long, cela pose les difficultés suivantes :

- il est difficile de trouver rapidement l'endroit où intervenir dans le fichier (c'est d'ailleurs le même constat avec les traitements de texte tels que Word ou Writer)
- le 1<sup>er</sup> affichage de la prévisualisation du rendu est plus lente dans l'IDE ou le navigateur
- les différentes parties du fichier ne peuvent pas être utilisées dans d'autres documents AsciiDoc (en tout cas, pas facilement)
- le décalage entre la position dans le texte du fichier AsciiDoc et la position correspondant dans le contenu prévisualisé est de plus en plus grand (surtout dans le cas où des fichiers tiers sont inclus). Cela est pénible à l'usage et ralenti le travail de rédaction.
- le travail à plusieurs sur le fichier est problématique (il faut travailler tour à tour).
- le partage du fichier conduit à partager l'intégralité du contenu alors que c'est souvent inutile

AsciiDoc offre une fonctionnalité très intéressante qui consiste à créer un fichier principal à partir de fichiers plus petits.

### 25.1.2. Quelle est la bonne "longueur" d'un fichier AsciiDoc ?

Si vous vous rendez compte que votre fichier AsciiDoc est "long", vous pouvez répartir son contenu dans différents fichiers AsciiDoc.

J'utilise le terme de "taille" pour être plus compréhensible mais ce n'est pas une histoire de taille.

Une bonne pratique consiste à avoir un fichier AsciiDoc par sujet de contenu.

Par exemple, dans le document que vous êtes en train de lire, j'ai un fichier AsciiDoc par chapitre. Parfois, un chapitre est même découpé en sous-fichiers !

L'idée est de ne pas avoir à traiter deux sujets différents dans un même fichier AsciiDoc. Autrement dit, un fichier ne devrait être modifié ou ne devrait évoluer que dans le cadre d'un sujet unique.

La difficulté est de déterminer les limites de son sujet. C'est l'expérience qui vous fera "ressentir" si le contenu de votre fichier AsciiDoc devrait être découpé en plusieurs fichiers indépendants.

Avec l'habitude, vous penserez même à créer un nouveau fichier AsciiDoc lorsque vous changerez de sujet.

## 25.2. Créer un document principal à partir de plusieurs autres

Vous avez écrit plusieurs fichiers AsciiDoc et vous souhaitez les réunir afin de générer un document pdf unique.

Si vous avez **limité la taille** de vos fichiers AsciiDoc en ne conservant qu'un sujet par fichier, vous allez avoir besoin de les regrouper dans un document unique.

Nous allons illustrer cette fonctionnalité à partir de 3 fichiers stockés dans le même répertoire `dossier_de_mon_support` :

```
\---dossier_de_mon_support
    sujet_A.adoc
    sujet_B.adoc
    sujet_C.adoc
```

Voici le contenu de chacun de ces fichiers :

- *contenu AsciiDoc du fichier sujet\_A.adoc*

```
//fichier sujet_A.adoc
= Titre principal du sujet A

== partie 1 du sujet A

== partie 2 du sujet A
```

- *contenu AsciiDoc du fichier sujet\_B.adoc*

```
//fichier sujet_B.adoc
= Titre principal du sujet B

== partie 1 du sujet B

== partie 2 du sujet B
```

- *contenu AsciiDoc du fichier sujet\_C.adoc*

```
//fichier sujet_C.adoc
= Titre principal du sujet C
```

```
== partie 1 du sujet C
```

```
== partie 2 du sujet C
```

L'objectif est d'avoir un fichier AsciiDoc qui regroupe ces trois fichiers.

La macro `include::[]` permet d'inclure un fichier. Cela est exactement la même chose que si vous copiez tout le contenu du fichier inclus et que vous le colliez dans le fichier depuis lequel l'inclusion est faite.

Il faut commencer par créer un fichier qui va inclure les autres fichiers. Ce fichier est nommé `mon_support_complet.adoc`.

Nous avons maintenant l'arborescence suivante :

```
\---dossier_de_mon_support
    mon_support_complet.adoc ①
    sujet_A.adoc ②
    sujet_B.adoc ②
    sujet_C.adoc ②
```

① fichier principal qui va inclure les autres fichiers

② sous-fichiers qui doivent être regroupés dans le fichier `mon_support_complet.adoc`

Pour distinguer facilement les fichiers inclus du fichier principal, je vous recommande fortement de préfixer le nom des sous fichiers par un underscore `_` :



```
\---dossier_de_mon_support
    mon_support_complet.adoc
    _sujet_A.adoc
    _sujet_B.adoc
    _sujet_C.adoc
```

Dans le fichier `mon_support_complet.adoc`, il faut inclure les autres fichiers :

```
//fichier mon_support_complet.adoc
= Titre du support complet ①
:toc: ②
:sectnums: ③

include::sujet_A.adoc[]
include::sujet_B.adoc[]
include::sujet_C.adoc[]
```

① Un titre principal est donnée au document complet

② activation de la table des matières (voir [partie 5.4.1, “Afficher la table des matières”](#))

③ activation de la numérotation des titres (voir [partie 5.4, “La table des matières”](#))

Puisque l'inclusion d'un fichier revient à copier-coller son contenu, nous pouvons imaginer que le fichier `mon_support_complet.adoc` contient le code AsciiDoc suivant :

```
//fichier mon_support_complet.adoc
= Titre du support complet
:toc:
:sectnums:

//fichier sujet_A.adoc
= Titre principal du sujet A

== partie 1 du sujet A

== partie 2 du sujet A
//fichier sujet_B.adoc
= Titre principal du sujet B

== partie 1 du sujet B

== partie 2 du sujet B
//fichier sujet_C.adoc
= Titre principal du sujet C

== partie 1 du sujet C

== partie 2 du sujet C
```

Si vous avez compris le chapitre sur [les titres](#), vous devez savoir qu'il ne peut y avoir qu'un seul titre principal marqué avec un seul caractère `=`. Or, dans le cas présent, le document final contient 4 titres de niveau 0 (titres principaux). Cela ne respecte pas la hiérarchie d'un document. La table des matières du fichier pdf généré **à partir du document principal** est d'ailleurs inadaptée :

# Table of Contents

Titre principal du sujet A .....	
1. partie 1 du sujet A .....	
2. partie 2 du sujet A .....	
Titre principal du sujet B .....	
1. partie 1 du sujet B .....	
2. partie 2 du sujet B .....	
Titre principal du sujet C .....	
1. partie 1 du sujet C .....	
2. partie 2 du sujet C .....	

Figure 120. table des matières avec plusieurs titres de niveau 0

La solution qui paraît évidente est de modifier à la main les niveaux de titre dans chaque sous fichier (ce qui peut être fastidieux et source d'erreurs si les sous-titres sont nombreux). Cependant, vous pouvez vouloir conserver dans chaque sous fichier la hiérarchie existante afin d'utiliser chacun d'eux de manière isolée.

La bonne solution est de modifier le niveau de titre via l'attribut `leveloffset`.

Cela peut être fait au niveau de la macro d'inclusion pour chaque fichier :

```
//fichier mon_support_complet.adoc
= Titre du support complet
:toc:
:sectnums:

\include::sujet_A.adoc[leveloffset=+1] ①
\include::sujet_B.adoc[leveloffset=+1] ①
\include::sujet_C.adoc[leveloffset=+1] ①
```

① tous les titres contenus dans le document inclus vont être augmentés d'un rang. Ainsi, un titre de niveau 0 devient un titre de niveau 1, un titre de niveau 1 devient un titre de niveau 2, etc.

# Table of Contents

1. Titre principal du sujet A .....
1.1. partie 1 du sujet A .....
1.2. partie 2 du sujet A .....
2. Titre principal du sujet B .....
2.1. partie 1 du sujet B .....
2.2. partie 2 du sujet B .....
3. Titre principal du sujet C .....
3.1. partie 1 du sujet C .....
3.2. partie 2 du sujet C .....

Figure 121. table des matières après modification du niveau des titres par fichier inclus

L'avantage de cette solution est qu'il est possible de définir par fichier la modification du niveau de ses titres. La modification du niveau n'impacte que le fichier concerné. `leveloffset=+2` signifie que les titres du fichier inclus doivent voir leur niveau actuel augmenté de 2 rangs. Ainsi, un titre de niveau 0 devient un titre de niveau 2, etc.

Il est possible de réduire le niveau des titres du fichier inclus en spécifiant une valeur négative.

 `leveloffset=-1` signifie que les titres du fichier inclus doivent voir leur niveau actuel diminué de 1 rang. Ainsi, un titre de niveau 1 devient un titre de niveau 0, etc. **Attention si le fichier inclus contient un titre de niveau 0. Il restera à zéro.**

Lorsque tous les fichiers inclus sont affectés par la même valeur de `leveloffset`, il peut être plus rapide de définir la valeur de `leveloffset` avant d'inclure les fichiers :

```
//fichier mon_support_complet.adoc
= Titre du support complet
:toc:
:sectnums:

:leveloffset: +1 ①

include::sujet_A.adoc[]
include::sujet_B.adoc[]
include::sujet_C.adoc[]

:leveloffset: -1 ②

//suite du contenu le cas échéant
```

① le niveau courant de titre est augmenté de 1. Tous les titres à venir seront traités avec leur

niveau de déclaration + 1.

- ② Il faut rétablir le niveau de titre "normal". C'est indispensable si l'on souhaite dans la suite du fichier, ajouter du contenu avec des niveaux de titre.



Utilisez avec prudence la dernière solution. Il faut garder en tête que la modification de niveau de titre impacte tout le document.

## 25.3. Adopter une structure efficace par document

Avec les logiciels de traitement de texte tels que Word ou Writer, un document donne un seul fichier. Je ne sais pas si vous le savez, mais cet unique fichier avec l'extension `docx` ou `odt` est en réalité un conteneur comme une archive peut l'être.

Prenez un fichier que vous avez créé avec votre traitement de texte et qui contient des images (sinon créez rapidement un fichier word ou writer avec du texte et des images). Faites-en une copie puis modifiez l'extension pour l'extension `zip` :

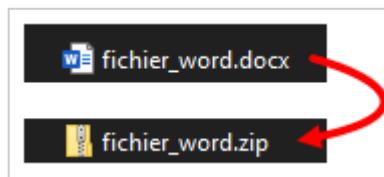
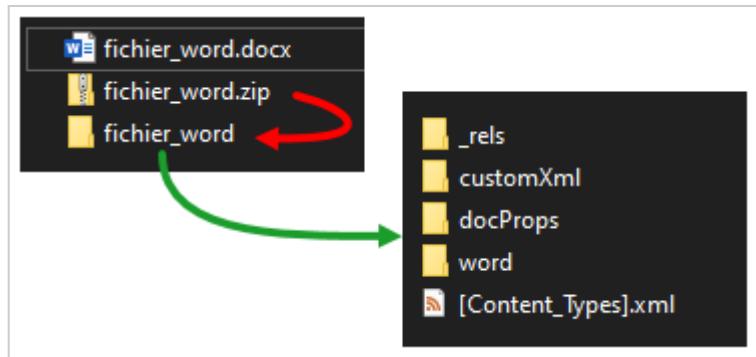


Figure 122. modification de l'extension d'un fichier word pour l'extension zip

Les fichiers avec l'extension `zip` sont des fichiers archive. Ce format permet de regrouper et de compresser plusieurs fichiers ou dossiers en un seul fichier.

Maintenant, si nous décompressons le fichier (clic droit sur le fichier puis "extraire tout..."), nous pouvons voir un dossier qui contient tous les dossiers et fichiers contenus dans le fichier `docx` :



Si le fichier contenait des images, vous les trouverez dans le dossier `word` et le sous-dossier `media`. Les autres fichiers contiennent du XML. Le fichier qui contient ce que vous avez écrit dans votre document word est le fichier `document.xml` situé dans le dossier `word`.

Finalement, un fichier de traitement de texte est tout simplement un dossier qui contient de nombreux fichiers permettant de gérer le document.

Bien sûr, cette gestion est totalement transparente car c'est le logiciel de traitement de texte qui organise tout ce petit monde.

Puisque vous êtes sur le point de délaisser votre traitement de texte, c'est à vous de faire ce travail de gestion des fichiers liés à un document.

Effectivement, lorsque vous rédigez un document avec AsciiDoc, vous pouvez rapidement avoir des fichiers un peu partout : des images, des sous-fichiers, des annexes, des documents que vous avez utilisés pour écrire votre support, etc.

Pour organiser tout ce petit monde, je vous propose l'organisation suivante pour chaque document AsciiDoc que vous allez écrire :



① fichier qui constitue votre document principal, c'est-à-dire celui à partir duquel le fichier pdf final sera généré. Je l'ai nommé `main` pour indiquer qu'il s'agissait du fichier principal mais vous pouvez lui donner le nom de votre choix.

② le dossier `assets` contient toutes les ressources qui devraient être communiquées avec le document généré une fois celui-ci terminé

③ le dossier `code` contient les fichiers de code dont sont extraits des régions de code illustrant le document (voir le chapitre [chapitre 15, Les blocs de code source](#))

④ le dossier `extras` contient toutes les ressources qui vont servir pour rédiger votre document (des recherches, des annexes, des vidéos, etc).

⑤ le dossier `images` contient les images utilisées de votre fichier asciidoc.

En adoptant cette structure dès à présent, vous prenez de bonnes habitudes en organisant toujours de la même façon chaque document.



Lorsque vous communiquez votre document au format AsciiDoc, il faut donner le document principal avec le dossier des images.

Finalement, si l'on compare avec un document réalisé avec un logiciel de traitement de texte, nous n'avons que le dossier `images` en plus. C'est d'ailleurs un avantage car vous pouvez remplacer une image directement dans le dossier des images sans avoir à ouvrir le document.

Vous pouvez modifier cette organisation mais si vous adoptez tous la même, le travail de groupe sur un même document est grandement facilité. Vous utilisez alors les mêmes chemins dans vos fichiers AsciiDoc et vous savez où chercher les éléments si vous travaillez avec le document de quelqu'un d'autre.



AsciiDoc s'attend (par défaut) à trouver les images au même niveau que le fichier AsciiDoc. Effectivement, le chemin d'une image est toujours relatif au fichier depuis

lequel elle est insérée. Ainsi, en utilisant un dossier `images` pour les images, l' insertion d'une image dans le fichier AsciiDoc se fera de la façon suivante :

```
image::images/nom_de_l_image.png[]
```

A ce titre :

- Le dossier `images` ne doit pas avoir un autre nom que `images`.
- Le dossier `images` doit être placé au même niveau que les fichiers AsciiDoc, donc ici au même niveau que le fichier `main.adoc`.

Cet emplacement par défaut peut être modifié mais je vous le déconseille fortement..

Si vous avez découpé votre document en sous-fichiers, veillez à garder les sous-fichiers au même niveau que le fichier principal sans quoi les chemins vers les fichiers inclus et les chemins vers les images ne seront plus valides (à moins de les adapter en conséquence puisqu'il s'agit de chemins relatifs).

Voici une organisation avec un document construit à partir de sous fichiers :

```
└── dossier_document_asciidoc
    ├── main.adoc ①
    ├── _sujet_A.adoc ②
    ├── _sujet_B.adoc ②
    ├── _sujet_C.adoc ②
    └── assets
        ├── code
        ├── extras
        └── images
```

① document principal AsciiDoc

② sous fichier AsciiDoc inclus dans le fichier principal

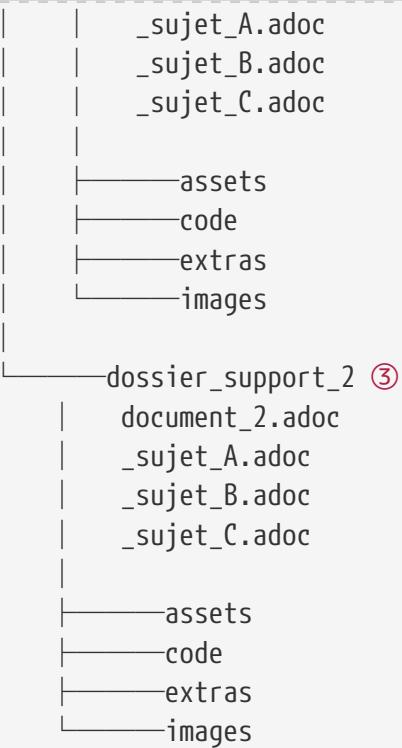


Pour identifier facilement les fichiers inclus dans le fichier principal, je les préfixe par un underscore `_`.

Il faut comprendre que pour chaque document AsciiDoc correspond un dossier avec la structure précédente.

Si vous avez plusieurs documents rangés dans un même répertoire parent, cela vous donne l'arborescence suivante :

```
└── mes_documents_asciidoc ①
    └── dossier_support_1 ②
        └── document_1.adoc
```



① dossier qui contient tous les documents AsciiDoc

② dossier correspondant à un document AsciiDoc. Il adopte la structure conseillée

③ un autre dossier correspondant à un document AsciiDoc. Il adopte la structure conseillée.

## 25.4. Partager un fichier pour ne pas se répéter

### 25.4.1. Partager un fichier qui ne contient pas d'images

Un des gros avantages d'AsciiDoc (encore un) est la possibilité d'inclure des fichiers depuis un autre fichier. Cette fonctionnalité peut être utilisée pour appliquer le **principe DRY** pour Don't Repeat Yourself soit en français "Ne vous répétez pas".

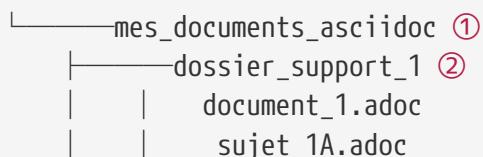
Si vous êtes amené à écrire la même chose que ce que vous avez déjà écrit dans un autre fichier, alors ne vous répétez pas et utiliser l'inclusion de fichier !

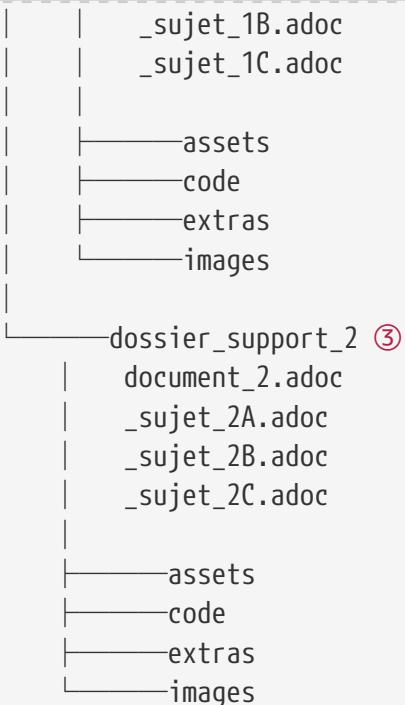
Voici une mise en contexte pour illustrer la réutilisation d'un fichier dans le cadre de l'organisation conseillée dans la [partie précédente](#).

Partons d'un dossier nommé `mes_documents_asciidoc`. Ce dossier contient tous les dossiers de travail AsciiDoc (c'est comme un dossier qui contiendrait vos fichiers Word ou Writer).

Chaque sous-dossier de ce dossier correspond à un document à part entière.

Cela donne l'arborescence suivante :





① dossier qui contient tous les documents

② dossier du premier document

③ dossier du second document

Imaginons que le fichier `document_2.adoc` soit composé du fichier `_sujet_2A.adoc`, `_sujet_2B.adoc` et `_sujet_2C.adoc`. Ce fichier nécessite également le même contenu que celui du fichier `_sujet_1A.adoc`. Nous pourrions dupliquer le fichier dans le dossier du document 2 mais cela nécessiterait de modifier les deux fichiers en cas de mise à jour. La bonne solution est d'inclure le fichier `_sujet_1A.adoc`.

Voici le contenu AsciiDoc du fichier `document_2.adoc` :

```
//fichier document_2.adoc
= Titre principal du document 2
//ici les attributs utiles (table des matières, numérotation des chapitres, etc.)

//inclusion des sous fichiers
include:::_sujet_2A.adoc[] ①
include:::_sujet_2B.adoc[] ①
include:::_sujet_2C.adoc[] ①

//inclusion du fichier situé dans le dossier du document 1
include:::./dossier_support_1/_sujet_1A.adoc[] ②
```

① sous fichiers situés au même niveau que le fichier courant (celui sur lequel nous travaillons)

② fichier situé dans un autre document. **Le chemin est relatif au fichier courant.** Il faut remonter d'un dossier via `..` puis entrer dans le dossier `dossier_suport_1` pour cibler le fichier `_sujet_1A.adoc`.



Attention, si les fichiers inclus contiennent des niveaux de titre, il faut veiller à ce que la hiérarchie de titre soit cohérente (voir [modifier les niveaux de titre par fichier ou pour un ensemble de fichiers](#))

Finalement, c'est très simple d'inclure un fichier qui serait situé dans un autre répertoire. Il suffit d'indiquer le chemin relatif vers ce fichier.



Si le fichier inclus contient lui-même des inclusions, vous n'avez rien de particulier à faire. Vous n'avez même pas à vous en soucier.

## 25.4.2. Partager un fichier qui contient des images

### Le cas de l'inclusion d'un fichier situé dans un autre dossier mais utilisant des images

Nous venons de voir qu'en incluant un sous fichier dans un autre, alors que ce sous fichier contient lui-même des inclusions vers d'autres sous fichiers, il n'y a rien à faire. AsciiDoc est capable de construire le document tout seul.

Mais, pour les images, il en va autrement. Si le sous fichier inclus utilise des images, le chemin vers les images sera relatif au fichier principal. Cela conduit AsciiDoc à ne pas trouver les images du fichier inclus.



Le chemin vers une image est toujours relatif au fichier principal.

Pour comprendre cette problématique, voici l'arborescence qui va servir mon propos :

```
└── mes_documents_asciidoc_pb_image
    ├── français
    │   ├── expression_ecrite.adoc
    │   ├── _la_phrase.adoc
    │   └── _la_ponctuation.adoc
    |
    └── histoire
        ├── traiter_sujet_histoire.adoc
        |
        └── images
            └── illustration_ponctuation.png
```

```
└── mes_documents_asciidoc_pb_image
    ├── français
    │   ├── expression_ecrite.adoc
    │   ├── _la_phrase.adoc
    │   └── _la_ponctuation.adoc
    |
    └── histoire
        ├── traiter_sujet_histoire.adoc
        |
        └── images
            └── photo_copie_histoire.png
```

Il y a un dossier **français** et un dossier **histoire** qui contiennent chacun les différents éléments d'un document.

Le document **traiter\_sujet\_histoire.adoc** utilise l'image issue de son dossier et mobilise le fichier rappelant la ponctuation :

## fichier traiter\_sujet\_histoire.adoc

= Comment traiter un sujet d'histoire ?

Voici la photo d'une copie d'histoire :

image::images/photo\_copie\_histoire.png[] ①

Voici un rappel à propos de la ponctuation :

include::../francais/\_la\_ponctuation.adoc[] ②

① le chemin de l'image est bien relatif au fichier courant

② le fichier sur la ponctuation est situé dans un autre dossier dont le chemin relatif indique qu'il faut remonter d'un dossier, puis rentrer dans le dossier `francais` pour cibler le fichier `_la_ponctuation.adoc`.

## fichier \_la\_ponctuation.adoc

La ponctuation est très importante !

Image illustrant l'intérêt de la ponctuation :

image::images/illustration\_ponctuation.png[] ①

① le chemin de l'image est bien relatif au fichier courant

Je vous ai dit à plusieurs reprises qu'inclure un fichier revient à copier-coller son contenu dans le fichier appelant.

On peut donc imaginer que le fichier `traiter_sujet_histoire.adoc` possède le contenu suivant après inclusion :

```
//fichier traiter_sujet_histoire.adoc  
= Comment traiter un sujet d'histoire ?
```

Voici la photo d'une copie d'histoire :

image::images/photo\_copie\_histoire.png[]

Voici un rappel à propos de la ponctuation :

```
//contenu inclus :  
La ponctuation est très importante !
```

Image illustrant l'intérêt de la ponctuation :

image::images/illustration\_ponctuation.png[] ①

① le chemin relatif ne mène pas à l'image qui était utilisée par le fichier inclus.

Comme vous l'aurez compris, les chemins relatifs des images utilisées dans les fichiers inclus ne sont plus corrects. Cela s'explique par le fait qu'AsciiDoc considère que le chemin relatif commence par défaut depuis le fichier courant.

Il y a deux solutions :

- copier l'image `francais/images/illustration_ponctuation.png` dans le dossier `histoire/images/`. Cependant, si l'image est mise à jour dans le premier dossier, elle doit l'être également dans le second. Cela est gérable pour une image, mais pour plusieurs, cela devient impossible à maintenir.
- modifier le chemin relatif des images utilisé par AsciiDoc avant d'inclure le fichier. Cela se fait en précisant le début du chemin relatif via l' attribut de document `imagesdir` :

```
= Comment traiter un sujet d'histoire ?
```

```
Voici la photo d'une copie d'histoire :
```

```
image::images/photo_copie_histoire.png[]
```

```
Voici un rappel à propos de la ponctuation :
```

```
:imagesdir: ../francais/ ①
```

```
include::../francais/_la_ponctuation.adoc[]
```

```
:imagesdir: ②
```

① on indique à AsciiDoc qu'il doit placer ce chemin avant tout chemin spécifié dans la macro `image::[]`. Il suffit de reprendre le chemin relatif qui mène au dossier du fichier à inclure.

② on indique à AsciiDoc qu'il doit placer une chaîne vide avant tout chemin spécifié dans la macro `image::[]`(c'est la valeur par défaut lorsque rien n'est spécifié).

Pour comprendre ce qu'il se passe, voici le fichier `traiter_sujet_histoire.adoc` avec le contenu du fichier inclus :

```
= Comment traiter un sujet d'histoire ?
```

```
Voici la photo d'une copie d'histoire :
```

```
image::images/photo_copie_histoire.png[]
```

```
Voici un rappel à propos de la ponctuation :
```

```
:imagesdir: ../francais/ ①
```

```
//début du contenu du fichier inclus
```

La ponctuation est très importante !

Image illustrant l'intérêt de la ponctuation :

```
image:.../francais/images/illustration_ponctuation.png[] ②  
//fin du contenu du fichier inclus
```

```
:imagesdir: ③
```

- ① il est indiqué à AsciiDoc qu'à partir de cette ligne, tous les chemins spécifiés dans la macro `image::` seront préfixés automatiquement par `../francais/`
- ② AsciiDoc a automatiquement ajouté la valeur de l'attribut `imagesdir` avant le chemin spécifié vers l'image. Le chemin relatif est maintenant `../francais/images/illustration_ponctuation.png` ce qui permet bien d'atteindre l'image située dans l'autre dossier.
- ③ il est indiqué à AsciiDoc qu'il ne doit plus rien mettre devant les chemins des images à partir de cette ligne.



Il ne faut pas oublier d'affecter une chaîne vide à l'attribut de document `imagesdir` après l'insertion du fichier. Sans cela, AsciiDoc continuera d'ajouter au début du chemin de chaque image insérée le chemin contenu dans cet attribut.

## 25.5. Versionner son travail

Un document AsciiDoc n'est qu'un simple fichier texte.

Il est très facile de le versionner avec un outil de gestion de version. Un tel outil permet de gérer différentes versions d'un même fichier, de restaurer des versions antérieures, de voir les modifications entre deux versions et bien d'autres choses encore.

Je ne développerai pas ce point ici, car un cours est dédié au versionnement (voir [\[book\\_versionner\\_ses\\_fichiers\]](#))

# 26. Le fichier de thème PDF

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 23:48 | Auteur : Emmanuel Ravrat

Durée de réalisation : 1h

## 26.1. Qu'est-ce qu'un fichier de thème PDF ?

Lorsque vous cliquez sur le bouton **[PDF]** de la barre d'outil du plugin AsciiDoc de votre IDE, un fichier pdf est généré avec un rendu par défaut.

Ce rendu par défaut est configuré dans un **fichier de thème**.

Si vous voulez voir à quoi ressemble un fichier de thème, voici un [lien vers le fichier de thème utilisé par défaut](#)

Le fichier de thème peut être utilisé entre autres choses pour :

- choisir la police à utiliser, sa taille et sa couleur
- configurer les marges
- configurer le contenu de l'entête et du pied de page
- configurer le rendu de certains éléments (titre, tableau, blocs, table des matières, entête de page, pied de page, etc.)

L'objectif n'est pas de vous former à la création ou à la modification profonde du thème par défaut. Cela nécessiterait des chapitres spécifiques et du temps. Cependant, pour ceux que cela intéresserait, vous pouvez lire la documentation à partir de [cette page](#).

 Le fichier de thème n'est pas utilisé par AsciiDoc mais par **Asciidoctor PDF**. C'est un outil capable de générer un fichier pdf à partir de code AsciiDoc.

C'est cet outil qui est utilisé par le plugin AsciiDoc de votre IDE.

## 26.2. Mettre en place son propre fichier de thème PDF

### 26.2.1. Placer le fichier de thème au même niveau que le fichier AsciiDoc

Pour pouvoir agir sur le rendu PDF, il faut créer un fichier de thème afin de remplacer celui qui est utilisé par défaut.

Nous allons partir de l'arborescence suivante :

```
└── mes_documents_asciidoc
    ├── français
    │   ├── expression_ecrite.adoc
    │   ├── _la_phrase.adoc
    │   └── _la_ponctuation.adoc
```

```
└── mes_documents_asciidoc_theme
    └── histoire
        ├── traiteur_sujet_histoire.adoc
        └── images
            └── illustration_ponctuation.png

    └── images
        └── photo_copie_histoire.png
```

Nous travaillons sur le document `traiteur_sujet_histoire.adoc` et nous générerons le fichier pdf depuis ce dernier.

Pour commencer, nous allons créer un fichier de thème **au même niveau** que le document AsciiDoc. Le fichier de thème doit avoir un nom qui se termine par `-theme.yml` (c'est une convention de nommage). Ecrivons le contenu suivant dans le fichier de thème nommé `custom-theme.yml` (L'extension `yml` indique que le fichier contient du YAML.) :

```
extends: default ①
base: ②
font-color: ff0000 ③
```

- ① la clé `extends` indique le nom du thème qu'il faut reprendre. Cela signifie "reprend toute la configuration du thème nommé "default" car nous allons l'étendre, c'est-à-dire ajouter ou modifier certaines valeurs"
- ② la clé `base` indique une configuration des paramètres fondamentaux du thème, c'est-à-dire que tous les éléments textuels sont impactés (liste, tableau,etc.)
- ③ la clé `font-color` permet de spécifier la couleur de la police, ici du rouge au format hexadécimal mais sans le caractère #.

L'arborescence est maintenant la suivante :

```
└── mes_documents_asciidoc_theme
    ├── franais
    │   ├── expression_ecrite.adoc
    │   ├── _la_phrase.adoc
    │   └── _la_ponctuation.adoc
    └── histoire
        ├── custom-theme.yml ①
        ├── traiteur_sujet_histoire.adoc
        └── images
            └── photo_copie_histoire.png
```

① le fichier de thème `custom-theme.yml` est placé dans le même répertoire que le fichier AsciiDoc.



L'organisation de l'arborescence reprend celle qui est présentée dans la partie [partie 25.3, "Adopter une structure efficace par document"](#) du chapitre [chapitre 25, "Les bonnes pratiques de création d'un document AsciiDoc".](#)

Voici le contenu du fichier `traiter_sujet_histoire.adoc` :

```
= La véritable histoire de France
```

Chuck Norris a toujours su rester dans l'ombre des grands qui ont fait l'histoire.

Voici quelques faits historiques qui ont été vérifiés :

\* Chuck Norris a été le vrai roi de France avant Louis XIV :

En 1643, avant même que Louis XIV ne devienne roi, Chuck Norris avait déjà mis en place un système de gouvernement tellement efficace que Louis XIV n'a eu qu'à faire une petite visite pour voir comment le travail se faisait. Chuck Norris n'a jamais vraiment été roi, mais il a enseigné à Louis XIV comment gouverner avec un simple regard.

\* ...

Nous avons un fichier de thème personnalisé mais Asciidoc ne le sait pas. Il faut le lui "dire" via l'attribut de document `pdf-theme` à déclarer au niveau des attributs d'entête (ceux qui sont déclarés sous le titre principal). Cet attribut reçoit le chemin relatif vers le fichier de thème à utiliser :

```
= La véritable histoire de France
```

```
:pdf-theme: ./custom-theme.yml ①
```

Chuck Norris a toujours su rester dans l'ombre des grands qui ont fait l'histoire.

① Chemin relatif vers le fichier de thème qui doit être utilisé. Ici, le chemin indique que le fichier de thème est au même niveau que le fichier AsciiDoc.



L'emplacement du fichier de thème est **relatif** au fichier depuis lequel la génération du pdf est effectuée.

Maintenant, il faut cliquer sur le bouton **[ PDF ]** pour générer le fichier PDF. Si le fichier de thème est bien pris en compte, la police est rouge :

# La véritable histoire de France

Chuck Norris a toujours su rester dans l'ombre des grands qui ont fait l'histoire.

Voici quelques faits historiques qui ont été vérifiés :

- Chuck Norris a été le vrai roi de France avant Louis XIV : En 1643, avant même que Louis XIV ne devienne roi, Chuck Norris avait déjà mis en place un système de gouvernement tellement

Figure 123. application du thème personnalisé avec une couleur de police rouge

Cette solution qui fonctionne très bien et permet de découvrir la base n'est pas du tout évolutive. Je vous conseille de lire le point suivant.

## 26.2.2. Un seul fichier de thème pour tous les fichiers asciidoc !

Placer le fichier de thème au même niveau que le fichier AsciiDoc est problématique. Si nous souhaitons générer le fichier pdf du fichier AsciiDoc situé dans le répertoire `francais` de notre arborescence, il faut créer un nouveau fichier de thème ou indiquer un chemin vers le fichier de thème que nous venons de créer. Ce ne sont pas des solutions qui facilitent la maintenance.

La meilleure solution est de placer le fichier de thème dans un dossier situé à la racine de tous les documents AsciiDoc. Voici l'arborescence mise à jour :

```

└── mes_documents_asciidoc_theme
    ├── .config ①
    │   └── themes ②
    │       └── custom-theme.yml ③
    ├── francais
    │   ├── expression_ecrite.adoc
    │   ├── _la_phrase.adoc
    │   ├── _la_ponctuation.adoc
    │   └── images
    │       └── illustration_ponctuation.png
    └── histoire
        ├── traiter_sujet_histoire.adoc
        └── traiter_sujet_histoire.pdf
        └── images
            └── photo_copie_histoire.png

```

① dossier qui va contenir toute la configuration de vos documents asciidoc (attributs partagés, fichiers de thèmes partagés, etc.). j'ai choisi arbitrairement le nom de dossier `.config` avec un `.` pour distinguer ce dossier des dossiers correspondant à des documents AsciiDoc.

② dossier qui va contenir les fichiers de thèmes (utiles si vous prévoyez des thèmes différents en fonction des destinataires ou de la nature de vos supports)

③ notre fichier de thème personnalisé

L'avantage de l'utilisation d'un fichier de thème placé dans un répertoire parent à tous les documents est que le chemin vers le fichier de thème sera toujours le même pour tous les documents AsciiDoc !

Voici le chemin vers le fichier de thème à utiliser dans tous les documents AsciiDoc :

```
= Le titre du document concerné  
:pdf-theme: ../../config/themes/custom-theme.yml ①
```

① Pour accéder au fichier de thème, Asciidoc doctor doit remonter d'un dossier puis entrer dans le dossier `.config` puis dans le dossier `themes` et utiliser le fichier `custom-theme.yml`.

Notre fichier `traiter_sujet_histoire.adoc` est modifié pour le résultat suivant :

```
= La véritable histoire de France  
:pdf-theme: ../../config/themes/custom-theme.yml
```

Chuck Norris a toujours su rester dans l'ombre des grands qui ont fait l'histoire.

Si nous générions un pdf à partir du document placé dans `mes_documents_asciidoc/francais/expression_ecrite.adoc`, le chemin serait exactement le même :

```
= L'expression écrite  
:pdf-theme: ../../config/themes/custom-theme.yml ①  
  
include::_la_phrase.adoc[]  
include::_la_punctuation.adoc[]
```

① chemin relatif vers le fichier de thème



Cette façon de faire implique de toujours garder la même organisation pour chacun des documents asciidoc.

Pour cela, suivez les recommandations de la partie partie 25.3, "Adopter une structure efficace par document".

Si vous souhaitez profiter d'un chargement encore plus flexible, lisez le point suivant.

### 26.2.3. Rendre variable le chemin vers le fichier de thème

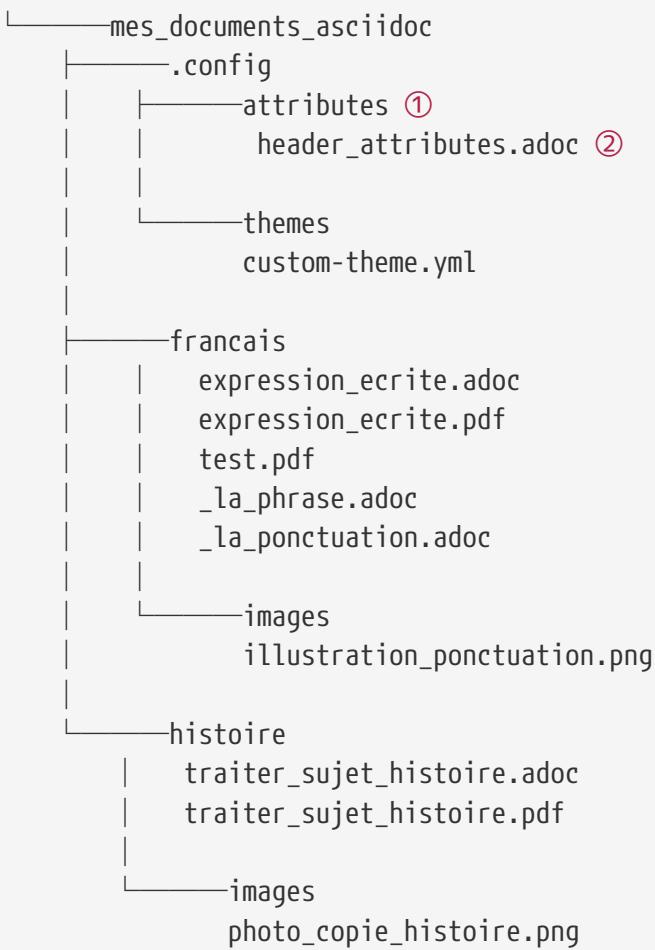
Si vous souhaitez utiliser un fichier de thème différent dans des documents existants, il faut changer le nom du fichier dans tous les documents concernés. Cette solution est bien évidemment source d'erreurs et d'oublis.

L'astuce consiste à utiliser un attribut pour y stocker le chemin vers le fichier de thème à utiliser.

Créons un nouveau dossier `.config/attributes/`. Dans ce dossier, nous allons ajouter un nouveau

fichier `header_attributes.adoc`.

L'arborescence est maintenant la suivante :



- ① dossier qui va contenir les fichiers dans lesquels vont être déclarés des attributs partagés entre tous les documents
- ② fichier qui va contenir des attributs utiles aux attributs d'entête, c'est-à-dire ceux qui sont déclarés sous le titre principal d'un document AsciiDoc.

Dans le fichier `header_attributes.adoc`, nous allons stocker le chemin vers le fichier de thème dans un attribut :

```
//fichier contenant les attributs utiles aux attributs de document déclarés sous le
titre principal de chaque document

//chemin vers le fichier de thème (à utiliser depuis un document AsciiDoc)
:path_to_pdf_theme_filename: ../../config/themes/custom-theme.yml ①
```

- ① l'attribut `path_to_pdf_theme_filename` contient le chemin relatif qui mène au fichier de thème.

Maintenant, pour utiliser cet attribut depuis n'importe quel fichier AsciiDoc de notre arborescence, il faut inclure le fichier des attributs et utiliser l'attribut `path_to_pdf_theme_filename`.

Voici la mise à jour du fichier `traiter_sujet_histoire.adoc` :

```
include:../../config/attributes/header_attributes.adoc[] ①
= La véritable histoire de France
:pdf-theme: {path_to_pdf_theme_filename} ②
```

Chuck Norris a toujours su rester dans l'ombre des grands qui ont fait l'histoire.

- ① Le fichier contenant l'attribut qui va être utilisé à la ligne 3 est inclus. Pour rappel, une inclusion est équivalent à copier-coller le contenu du fichier inclus.
- ② L'attribut `pdf-theme` reçoit la valeur de l'attribut `path_to_pdf_theme_filename`.

Avec cette solution, si le nom du fichier de thème (voire son emplacement) vient à changer, il n'y a qu'une seule modification à faire depuis le fichier `header_attributes.adoc`.

Si vous utilisez différents fichiers de thème en fonction de vos supports, lisez le point suivant.

#### 26.2.4. Rendre variable le nom du fichier de thème

Si vous souhaitez pouvoir changer le nom du fichier de thème à utiliser pour un fichier, cela est possible en utilisant deux attributs : le premier va stocker le chemin relatif vers le dossier des thèmes et le second va stocker le nom du fichier de thème à utiliser par défaut :

Voici le fichier `header_attributes.adoc` mis à jour :

```
//fichier contenant les attributs utiles aux attributs de document déclarés sous le
titre principal de chaque document
//chemin vers le dossier des thèmes
:path_to_folder_pdf_theme: ../../config/themes/ ①
//nom du fichier de thème à utiliser
:pdf_theme_filename: custom-theme.yml ②
```

- ① un attribut personnalisé contient le chemin vers le dossier des fichiers de thème  
② le fichier de thème à utiliser est stocké dans l'attribut personnalisé `pdf_theme_filename`.

Rappel : Il ne doit pas y avoir de ligne vide dans le fichier `header_attributes.adoc` puisqu'il s'agit de l'entête du document.



Si vous laissez une ligne vide, le rendu devrait échouer ou vous devriez voir un problème dans le contenu affiché, c'est-à-dire quelque chose qui n'est pas logique, inattendu.

Maintenant, pour utiliser le thème `custom-theme.yml` depuis le fichier `traiter_sujet_histoire.adoc`, voici comment faire :

```
include:../../config/attributes/header_attributes.adoc[] ①
= La véritable histoire de France
:pdf-theme: {path_to_folder_pdf_theme}{pdf_theme_filename} ②
```

Chuck Norris a toujours su rester dans l'ombre des grands qui ont fait l'histoire.

- ① Le fichier contenant l'attribut qui va être utilisé à la ligne 3 est inclus. Pour rappel, une inclusion est équivalent à copier-coller le contenu du fichier inclus.
- ② L'attribut `pdf-theme` reçoit la concaténation de la valeur de l'attribut `path_to_folder_pdf_theme` et celle de l'attribut `pdf_theme_filename`.

Ainsi, vous utilisez par défaut le fichier `custom-theme.yml`. Mais si vous voulez utiliser un autre fichier de thème qui serait placé dans le dossier des thèmes, voici comment procéder :

```
include:../../config/attributes/header_attributes.adoc[] ①
= La véritable histoire de France
:pdf_theme_filename: autre_fichier_de_theme.yml ②
:pdf-theme: {path_to_folder_pdf_theme}{pdf_theme_filename} ③
```

- ① récupération du chemin vers le dossier de thème et du nom du fichier de thème à utiliser
- ② le nom du fichier de thème `autre-fichier-de-theme.yml` écrase la valeur `custom-theme.yml` qui avait été préalablement affectée à `pdf_theme_filename`

Pour rappel, il ne faut pas laisser de ligne vide entre les attributs déclarés sous le titre et la première ligne de texte du fichier AsciiDoc.

Ceci ne fonctionnera pas :

```
include:../../config/attributes/header_attributes.adoc[]
= La véritable histoire de France
:pdf_theme_filename: autre_fichier_de_theme.yml
①
:pdf-theme: {path_to_folder_pdf_theme}{pdf_theme_filename}
```

Chuck Norris a toujours su rester dans l'ombre des grands qui ont fait l'histoire.

- ★
- ① une ligne est laissée vide avant l'attribut de document `pdf-theme`. Comme il est précisé dans le chapitre [chapitre 19, Les attributs de document](#), les attributs de documents doivent être déclarés sous le titre principal sans laisser de ligne vide.

# 27. Mise en forme à l'aide de rôles personnalisés

Version 1.0.0 | Dernière mise à jour : 26/08/2024 à 23:54 | Auteur : Emmanuel Ravrat

Durée de réalisation : 40min

## 27.1. Rappel sur les possibilités intégrées de formatage du texte

Pour formater le texte, il existe les caractères de formatage `\`, `*`, `_`, `#`, `^`, `~` (voir [chapitre 7, Mise en forme du texte](#)).

AsciiDoc prévoit également des rôles intégrés qui permettent d'appliquer un formatage complémentaire.

```
[.small]#Ceci est du texte écrit plus petit# ①
```

① le rôle `small` est placé entre crochets. Le nom du rôle est précédé d'un point.

Plusieurs rôles intégrés sont disponibles `lead`, `big`, `small`, `underline`, `line-through`, `text-left`, etc. (voir le chapitre [chapitre 8, Mise en forme avec les rôles intégrés](#))

## 27.2. Créez un rôle personnalisé

Un **rôle personnalisé** permet d'étendre les possibilités de mise en forme. Un rôle est un nom associé à un formatage de texte **que vous définissez vous-même**.

Il est possible d'appliquer un rôle à un paragraphe complet ou à un morceau de texte. Un rôle est appliqué en précédant le contenu à formater du nom du rôle encadré de crochets `[]`. Le nom du rôle est précédé d'un point (même si ce n'est pas obligatoire).

Lorsqu'un rôle doit formater un morceau de texte, ce dernier doit être encadré de `#`.

```
[.chuck]#Chuck n'aime que le rôle 'chuck'# ①
```

① Le rôle `chuck` n'existe pas nativement dans le langage. Il faut le créer ce que nous allons faire immédiatement.

Pour créer un rôle personnalisé, il faut savoir créer un fichier de thème et le lier à son document AsciiDoc. Si ce n'est pas votre cas, il faut impérativement lire le chapitre [chapitre 26, Le fichier de thème PDF](#).

Une fois le fichier de thème créé, celui-ci doit avoir au minimum le contenu suivant :

```
extends: default ①
```

- ① Cela indique qu'il faut reprendre le thème utilisé par défaut. Ainsi, il est possible d'ajouter ou de modifier certains paramètres du thème par défaut.

Pour ajouter le formatage de rôles personnalisés dans le fichier de thème, il faut ajouter une clé `role` au même niveau que la clé `extends`, c'est-à-dire contre la marge :

```
extends: default  
role: ①
```

- ① cette clé `role` va contenir tous les rôles personnalisés que vous allez créer.

Ensuite, il faut choisir le nom du rôle (pour nous, ce sera `chuck`) et configurer le formatage à appliquer à ce rôle :

```
extends: default  
role:  
  chuck: ①  
    font-color: c00c00 ②  
    font-size: 2rem ③  
    border-width: 1 ④  
    background-color: fdd56c ⑤
```

- ① nom du rôle **décalé d'une tabulation** par rapport à la marge pour indiquer qu'il est enfant de la clé `role`
- ② couleur de la police rougeâtre avec la clé `font-color` **décalée de deux tabulations** par rapport à la marge pour indiquer qu'elle est enfant de la clé `chuck`. La valeur est une valeur hexadécimale. Le caractère # de la couleur `#c00c00` peut être omis.
- ③ La taille de police avec la clé `font-size` sera deux fois la taille de police de base.
- ④ bordure autour du texte formaté (changer la valeur n'a pas d'effet dans une sortie pdf)
- ⑤ couleur de fond

## Chuck n'aime que le rôle `chuck`

Figure 124. rendu d'un texte formaté avec le rôle `chuck` :

Pour rappel, un rôle peut être appliqué à un paragraphe entier.

```
[.chuck] ①
```

À la préhistoire, Chuck Norris n'était pas un simple homme des cavernes ; il était le roi des cavernes, le maître des mammouths et le cauchemar des tigres à dents de sabre. On raconte que ses abdos étaient si sculptés qu'ils faisaient tomber les stalactites par pure admiration, et qu'une fois, il a convaincu un volcan de ne plus jamais exploser, juste pour lui faire plaisir.

Quand il s'attaquait à la chasse, les dinosaures avaient déjà pris leur retraite, et les pierres se transformaient en outils juste en entendant son nom.  
Chuck Norris a tellement impressionné les étoiles qu'elles ont décidé de briller encore plus fort rien que pour lui !

① Le rôle est appliqué à un paragraphe.

**À la préhistoire, Chuck Norris n'était pas un simple homme des cavernes ; il était le roi des cavernes, le maître des mammouths et le cauchemar des tigres à dents de sabre. On**

Figure 125. rendu du paragraphe avec le rôle `chuck`

Comme vous pouvez le voir, il n'est pas possible d'appliquer la même mise en forme sur un paragraphe complet et sur un morceau de texte avec le même rôle. Certaines propriétés ne peuvent pas être modifiées au niveau d'un paragraphe et d'autre au niveau d'un morceau de texte.

Si vous avez d'autres rôles, vous pouvez les déclarer les uns sous les autres dans le fichier de thème. Voici par exemple le rôle `jcvd` ajouté au fichier de thème :

```
extends: default
role:
  chuck:
    font-color: c00c00
    font-size: 2rem
    border-width: 1
    background-color: fdd56c
  jcvd: ①
    font-color: 007bff      # Bleu vif, en référence à son style dynamique et
                           énergique ②
    font-size: 1.8rem       # Taille légèrement plus petite pour souligner la
                           distinction
    border-width: 2         # Bordure un peu plus épaisse pour représenter sa
                           robustesse
    background-color: f5f5f5 # Gris clair, plus subtil pour un style élégant
    font-weight: bold       # Pour accentuer le dynamisme et la présence
    text-transform: uppercase # Pour refléter la force et l'impact de ses
                           performances
```

① le nouveau rôle `jcvd` est placé sous la clé `role`, soit à une tabulation de la marge de gauche.

② il est possible d'utiliser des commentaire dans un fichier YAML. Un commentaire commence par un `#`

Utilisation des rôles :

[.jcvd]#Jean-Claude Van Damme a réussi à casser une planche avec un coup de pied,#  
mais [.chuck]#Chuck Norris a fait en sorte que les planches se brisent d'elles-mêmes  
juste en les regardant#.

JEAN-CLAUDE VAN DAMME A RÉUSSI À CASSER UNE  
PLANCHE AVEC UN COUP DE PIED, mais Chuck Norris a  
fait en sorte que les planches se brisent d'elles-  
mêmes juste en les regardant

Figure 126. rendu des deux rôles utilisés

Si vous voulez un aperçu des propriétés de mise en forme, vous pouvez vous inspirer des propriétés définies dans le thème par défaut.

# 28. L'entête et le pied de page du pdf

Version 1.0.0 | Dernière mise à jour : 27/08/2024 à 00:09 | Auteur : Emmanuel Ravrat  
Durée de réalisation : 1h

## 28.1. Rendu par défaut

Pour voir le rendu pdf par défaut en termes d'entête et de pied de page, il faut commencer par créer un nouveau document AsciiDoc :

```
= JCVD déclare son admiration à Chuck ! ①
:doctype: book ②
:toc: ④
:sectnums: ⑤

== Déjà à 5 ans ! ③

== A l'adolescence ! ③

== A l'âge adulte... ③
```

① titre principal du document. Avant cette ligne, rien ne doit être affiché, pas même une ligne vide. (la déclaration d'un attribut ne revient pas à afficher quelque chose, donc c'est permis).

② le doctype `book` va forcer les parties de niveau 1 à se placer sur une nouvelle page. (voir le [chapitre 22, Le type de document](#)) Ces parties deviennent des chapitres.

③ chapitres constituant le document

④ affichage de la table des matières

⑤ activation de la numération des parties



Vérifiez bien que le doctype est `book` sans quoi la suite du cours ne va pas correspondre au résultat que vous allez obtenir.

Cliquez sur le bouton [ PDF ] de la barre d'outil AsciiDoc.

Observations :

- il y a une page de garde qui reprend le titre du document
- la table des matières est affichée et les titres sont numérotés
- chaque partie introduite par un titre de niveau 1 (soit un titre déclaré avec `==`) débute sur une nouvelle page.
- les numéros de page sont affichés en bas à droite ou à gauche en fonction de la page recto ou verso.

## 28.2. Compléter les métadonnées (auteur, date de révision, ...)

Les **métadonnées** sont des informations qui décrivent ou fournissent des détails supplémentaires sur le contenu du document. Elles ne font pas partie du contenu principal mais ajoutent un contexte essentiel, comme des informations sur l'auteur, la date de publication, la version, l'email, ou des détails spécifiques au document.

Les métadonnées se définissent dans l'entête du document AsciiDoc, c'est-à-dire sous le titre principal. Elles sont affichées sur la page de garde dans le fichier pdf.

Nous allons définir l'auteur, le numéro de version du document, la date de révision.

Voici les attributs permettant de définir ces métadonnées :

```
= JCVD déclare son admiration à Chuck !
```

```
:doctype: book
```

```
:toc:
```

```
:sectnums:
```

```
:author: Emmanuel Ravrat ①
```

```
:email: une_adresse@mail.fr ②
```

```
:revdate: 01/08/2024 à 12:18 ③
```

```
:revnumber: 2 ④
```

```
== Déjà à 5 ans !
```

```
== A l'adolescence !
```

```
== A l'âge adulte...
```

① Prénom et nom de l'auteur (**dans cet ordre !**)

② email de l'auteur

③ date de révision du document

④ numéro de version du document

Rien qu'avec ces métadonnées, la page de garde n'est plus la même. Générez le fichier pdf via le bouton **[ PDF ]** :

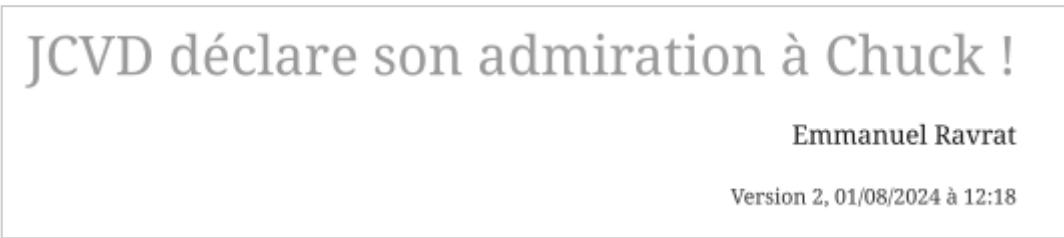


Figure 127. rendu de la page de garde avec les métadonnées

Le numéro de version est préfixé par le mot **Version**. Ce label est personnalisable. Il suffit de déclarer l'attribut **version-label** sous le titre avec la valeur de son choix :

```
= titre principal  
//ici des attributs d'entête  
:version-label: v. ①
```

① Le numéro de version sera maintenant préfixé par **v.**

Si vous travaillez à plusieurs sur un document, vous pouvez spécifier les auteurs de la façon suivante :

```
:author: Emmanuel Ravrat ; Chuck Norris ①
```

① Les auteurs sont séparés par un **;**

Et si vous souhaitez ajouter l'email de chaque auteur, cela se fait comme ceci :

```
:author: Emmanuel Ravrat <une_adresse@mail.fr>; Chuck Norris <chuck@norrisfr> ①
```

① Les adresses mail sont placées après chaque auteur entre **<** et **>**

## JCVD déclare son admiration à Chuck !

Emmanuel Ravrat <une\_adresse@mail.fr>; Chuck Norris <chuck@norrisfr>

Version 2, 01/08/2024 à 12:18

Figure 128. rendu de la page de garde avec de multiples auteurs

### 28.3. Personnaliser l'entête de page du fichier pdf

Nous allons personnaliser l'**entête de page** du fichier pdf.

Il faut dans un premier temps lier le fichier AsciiDoc à un fichier de thème pdf (si vous ne savez pas comment faire, lisez le chapitre [chapitre 26, Le fichier de thème PDF](#))

Dans notre mise en œuvre, le fichier de thème sera placé au même niveau que le document AsciiDoc (ce n'est pas **la technique recommandée**).

Voici le fichier de thème **my-theme.yml** qui va être lié à notre document AsciiDoc :

```
extends: default ①
```

① indique que la mise en page du thème par défaut doit être reprise en intégralité (cela évite de devoir partir de zéro).

Maintenant, il faut lier le fichier de thème au fichier AsciiDoc :

```
= JCVD déclare son admiration à Chuck !  
:doctype: book  
:pdf-theme: ./my-theme.yml ①  
:toc:  
:sectnums:  
:author: Emmanuel Ravrat  
:email: une_adresse@mail.fr  
:revdate: 01/08/2024 à 12:18  
:revnumber: 2  
  
== Déjà à 5 ans !  
  
== A l'adolescence !  
  
== A l'âge adulte...
```

① l'attribut `pdf-theme` est affecté du chemin relatif vers notre fichier de thème.

Maintenant, il ne reste plus qu'à configurer les paramètres d'affichage de l'entête de page dans le fichier de thème.

Il faut commencer par définir la hauteur de l'entête sans quoi il reste "désactivé".



```
extends: default  
header:  
    height: 15mm ①
```

① hauteur de l'entête en millimètres

Une fois que la hauteur de l'entête est définie, nous pouvons passer aux choses sérieuses.

Nous allons utiliser la clé `header` qui concerne toute la configuration de l'entête de page d'un fichier pdf. Cette clé contient les clés `recto` et `verso` pour mettre en forme les faces éponymes.

```
header:  
    height: 15mm  
    recto: ①  
        center:  
            content: "-- du texte centré dans l'entête au recto --" ②  
    verso: ③  
        center:  
            content: "-- du texte centré dans l'entête au verso --" ④
```

① configuration du rendu de l'entête face recto

② contenu à afficher dans l'entête face recto. Il sera centré car placé sous la clé `center`.

③ configuration du rendu de l'entête face verso

④ contenu à afficher dans l'entête face verso. Il sera centré car placé sous la clé `center`.

Générez le fichier pdf à partir du bouton [ PDF ].

— du texte centré dans l'entête au recto —

# Chapter 1. Déjà à 5 ans !

Figure 129. entête de page côté recto

— du texte centré dans l'entête au verso —

# Chapter 2. A l'adolescence !

Figure 130. entête de page côté verso

Il est possible de référencer le titre du chapitre ou d'une sous-partie dans l'entête :

```
header:
  height: 15mm
  recto: ①
    center:
      content: "-- {section-or-chapter-title} --" ②
  verso: ③
    center:
      content: "-- {section-or-chapter-title} --" ④
```

① configuration de l'entête côté recto

② configuration de l'entête côté verso (qui peut être différent du recto)

— Chapter 1. Déjà à 5 ans ! —

# Chapter 1. Déjà à 5 ans !

Figure 131. entête avec référence au titre du chapitre ou d'une sous-partie

Vous pouvez remplacer l'attribut `section-or-chapter-title` par les valeurs suivantes (je n'ai listé que celles qui me paraissent intéressantes) :

- `page-count` : numéro de page le plus élevé
- `page-number` : page en cours
- `page-layout` : mode d'affichage de la page courante (portrait ou paysage)
- `document-title` : titre principal du document
- `chapter-title` : titre du chapitre (correspond au titre de niveau 1 `==`) dans un document de type `book`
- `chapter-numeral` : numéro du titre du chapitre

- **section-title** : titre de section, c'est-à-dire le titre de niveau 2 déclaré avec ===
- **section-or-chapter-title** : titre de niveau 2 ou 1 en fonction de sa présence ou non sur la page courante.

Vous pouvez consulter la [liste complète dans la documentation](#).

Vous pouvez dans le fichier de thème, au niveau des clés **content**, référencer n'importe quel attribut déclaré dans le document, même un attribut de l'utilisateur (voir la [partie 19.4, "Les attributs de l'utilisateur"](#)), c'est-à-dire un attribut créé par vous-même.



```
header:  
  height: 15mm  
  recto:  
    center:  
      content: "{mon_attribut_personnalise}" ①
```

① utilisation d'un attribut déclaré dans le document.

Voici un autre exemple d'entête et de son rendu :

```
header:  
  height: 11mm  
  border-width: 1  
  vertical-align: bottom  
  padding: [0,0,6,0]  
  recto:  
    left:  
      content: "{chapter-title}"  
  verso:  
    right:  
      content: "{chapter-title}"
```

Chapter 1. Déjà à 5 ans !

## Chapter 1. Déjà à 5 ans !

Figure 132. rendu d'un pied de page avec contenu, bordure et positionnement du contenu

Toutes les clés utilisables sous la clé **header** dans le fichier de thème sont listées dans [cette page de la documentation](#). Vous trouverez entre autres la couleur de fond, la couleur de bordure, l'image de fond, la police, les marge, etc.

Si vous voulez en savoir davantage sur les possibilités de personnalisation des entêtes et pied de page, voici des parties de la documentation qui sont intéressantes :

- ↗ ajouter une image dans l'entête et / ou le pied de page
- ↗ écrire l'entête ou le pied de page sur plusieurs lignes
- ↗ désactiver l'entête et/ou le pied de page

## 28.4. Personnaliser le pied de page du fichier pdf

La personnalisation du **pied de page** est identique à la personnalisation de l'entête de page ([partie 28.3, “Personnaliser l'entête de page du fichier pdf”](#)).

Voici ce qui diffère :

- les clés qui sont utilisables sous la clé **footer** (voir la [liste sur la page de la documentation](#)). Ce sont très sensiblement les mêmes que pour le pied de page.

Ce qui nous intéresse plus particulièrement est la **pagination** et la mention de l'auteur.

Voici un exemple qui affiche le nom de l'auteur dans le pied de page à droite sur la page recto et à gauche sur la page verso. La pagination sera affichée sur la gauche face recto et à droite au verso (comme un livre)

```
extends: default
header:
  height: 15mm
  border-width: 1
  recto:
    left:
      content: "{chapter-title}"
  verso:
    right:
      content: "{chapter-title}"
footer:
  height: 15mm
  recto:
    left:
      content: "{page-number} / {page-count}"
    right:
      content: "{author}"
  verso:
    left:
      content: "{author}"
    right:
      content: "{page-number} / {page-count}"
```



Figure 133. rendu d'une page côté recto

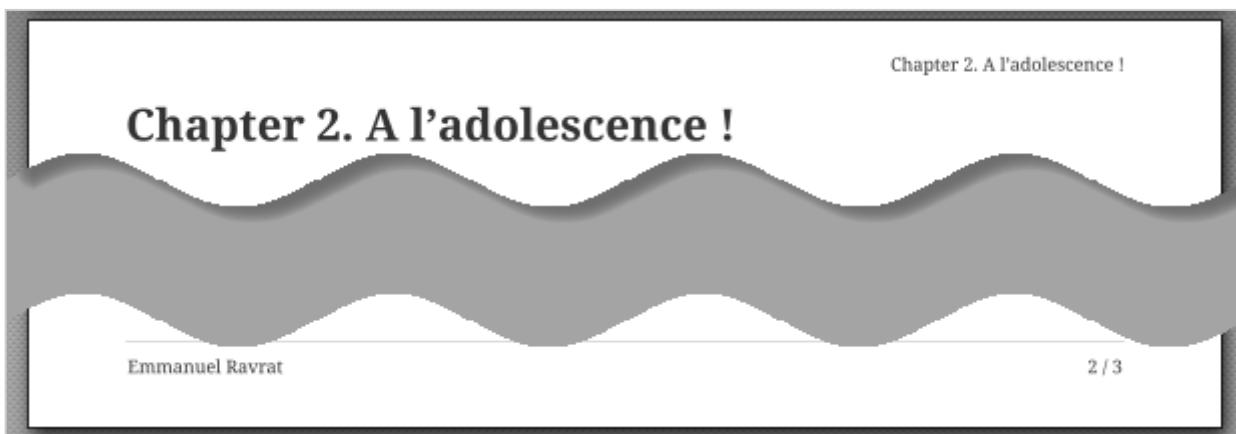


Figure 134. rendu d'une page côté verso

La pagination peut être désactivée avec l'attribut de document `pagenums` :



```
= Titre du document  
//ici des attributs d'entête  
:!pagenums: ①
```

① désactivation de la pagination

# 29. La page de garde

Version 1.0.0 | Dernière mise à jour : 27/08/2024 à 00:31 | Auteur : Emmanuel Ravrat

Durée de réalisation : 1h

## 29.1. La page de garde par défaut

Une **page de garde** est la première page d'un document de type "livre". C'est-à-dire que le document est composé de plusieurs chapitres entre autres choses.

Sur une page de garde, on peut s'attendre à trouver les informations suivantes :

- titre du document
- sous-titre le cas échéant
- nom de l'auteur ou des auteurs
- numéro de version
- date de publication
- logo ou image
- des informations juridiques (niveau de confidentialité, licences, etc.)
- etc.

Voici le fichier AsciiDoc qui va nous servir de support :

```
= JCVD déclare son admiration à Chuck !
:doctype: book ①
:pdf-theme: ./my-theme.yml ②
:toc:
:sectnums:
:author: Emmanuel Ravrat
:email: une_adresse@mail.fr
:revdate: 01/08/2024 à 17:57
:revnumber: 7

== Déjà à 5 ans !

== A l'adolescence !

== A l'âge adulte...
```

① Le type de document doit être **book**

② le document utilise un fichier de thème.



Pour qu'une page de garde soit générée, l'attribut de document **doctype** doit avoir la valeur **book**.

Pour forcer l'utilisation de la page de garde pour d'autres types de documents (par exemple avec le doctype `article`), il faut activer l'attribut de document `title-page` dans l'entête du document, c'est-à-dire sous le titre principal :

```
= Titre principal du document  
:doctype: article ①  
:title-page: ②
```

① le doctype `article` ne génère pas de page de garde, c'est le comportement par défaut.

② l'activation de l'attribut `title-page` force la génération d'une page de garde.

Ce fichier contient des métadonnées (voir [partie 28.2, “Compléter les métadonnées \(auteur, date de révision, ...\)](#)) (auteur, email, date de mise à jour, numéro de version). Ces métadonnées sont déclarées grâce à des attributs de document placés dans l'entête du document, c'est-à-dire immédiatement sous le titre.

Ce sont ces métadonnées qui sont reprises sur la page de garde.

Générez le fichier pdf à partir du bouton [ PDF ] de la barre d'outils AsciiDoc.

# JCVD déclare son admiration à Chuck !

Emmanuel Ravrat

Version 7, 01/08/2024 à 17:57

Figure 135. rendu de la page de garde avec style par défaut

Le contenu est centré verticalement et aligné sur la droite. Ce rendu est mis en forme par le thème utilisé par défaut.

L'objectif est de personnaliser cette page de garde en répondant au besoin le plus courant :

- Ajouter un logo
- Ajouter une image de fond
- Changer la couleur de police

## 29.2. Personnaliser la page de garde

### 29.2.1. Ajout d'un logo à la page de garde

J'ai un joli logo "Chuck Norris" pour la page de garde, son chapeau de cowboy :

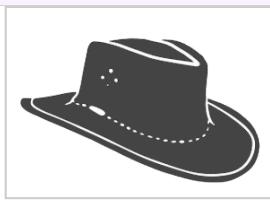


Figure 136. logo qui va être utilisé sur la page de garde

L'ajout d'un logo se fait via l'attribut de document `:title-logo-image`. Si vous ne savez pas ce qu'est un attribut de document, vous pouvez lire le chapitre [chapitre 19, Les attributs de document](#))

Cet attribut est déclaré dans l'entête du document, c'est-à-dire immédiatement sous le titre principal :

```
= JCVD déclare son admiration à Chuck !
:doctype: book
:pdf-theme: ./my-theme.yml
:toc:
:sectnums:
:author: Emmanuel Ravrat
:email: une_adresse@mail.fr
:revdate: 01/08/2024 à 17:57
:revnumber: 7
:title-logo-image: image:images/logo_chuck.png[] ①

== Déjà à 5 ans !

== A l'adolescence !

== A l'âge adulte...
```

① appel de l'image `logo_chuck.png` via un chemin relatif. Le dossier `images` est donc placé au même niveau que le document AsciiDoc.

Générez le fichier pdf via le bouton **[ PDF ]**.

Le logo est placé à droite (comme le texte). Il peut être déplacé en spécifiant des attributs de position et de dimensionnement :

```
= JCVD déclare son admiration à Chuck !
//...
:title-logo-image: image:images/logo_chuck.png[top=0] ①
```

① l'attribut `top` permet de spécifier le décalage par rapport au bord haut de la zone de contenu.



Figure 137. positionnement du logo par défaut et spécifié

Nous allons finalement placer le logo en bas à gauche :

```
:title-logo-image: image:images/logo_chuck.png[top=20cm,align=left] ①
```

① pour déterminer la distance entre le bord haut et le logo, j'ai utilisé des cm. Comme je sais que le format est A4 (soit 21 x 29.7cm), en plaçant à 20cm du haut, je voyais à peu près où allait être placé le logo. L'attribut `align` accepte les valeurs `right`, `center` et `left`.

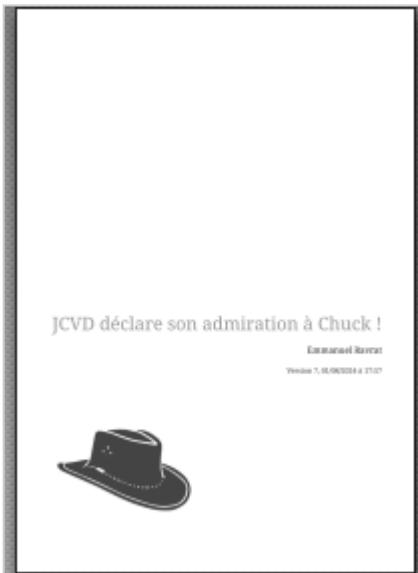


Figure 138. positionnement du logo en bas à gauche

Enfin, la largeur `width` et la hauteur `height` de l'image peuvent être précisées sous la forme d'un entier **sans unité**. (l'unité prise en compte est le pixel).

```
:title-logo-image: image:images/logo_chuck.png[top=22cm,align=left,width=250] ①
```

① La largeur de l'image est fixée à 250 pixels



Si une image de logo est définie dans le fichier de thème, c'est l'image de logo définie via l'attribut `title-logo-image` qui sera prioritaire.

## 29.2.2. Ajout d'une image de fond sur la page de garde

Pour définir une **image de fond** sur la page de garde, il faut définir l'attribut `title-page-background-image` :

```
:pdf-theme: ./my-theme.yml
//... autres attributs
:title-logo-image: image:images/logo_chuck.png[top=22cm,align=left,width=250]
:title-page-background-image:
image:images/image_de_fond_pour_page_de_garde.png[position=top] ①
```

① l'image de fond est appelée via la macro en ligne `image:[]` en lui précisant le chemin relatif vers celle-ci.



Figure 139. rendu de la page de garde avec *image de fond*

L'image est centrée verticalement par défaut et est adaptée aux limites de la page.

L'image peut être positionnée en haut (`top`), en bas (`bottom`) ou au centre (`center`) qui est la valeur par défaut.

Voici comment positionner l'image en haut avec la valeur d'attribut `position` à `top` :

```
:title-page-background-image:
image:images/image_de_fond_pour_page_de_garde.png[position=top]
```

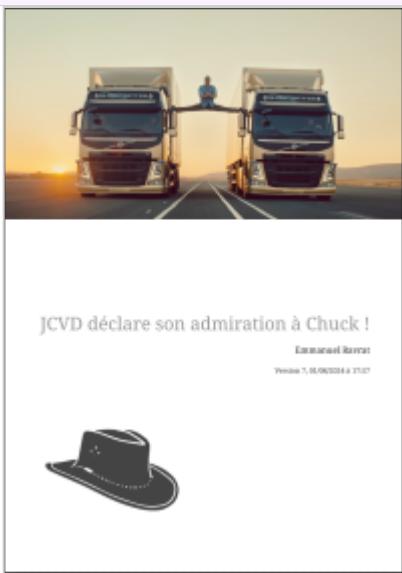


Figure 140. image de fond positionnée en haut de la page de garde



L'image de logo est toujours positionnée au-dessus de l'image de fond. C'est-à-dire que si l'image de fond était placée en bas de la page, le logo resterait visible.

L'adaptation de l'image à la largeur de la page peut être modifiée. Par défaut, l'image s'adapte à la largeur. Si elle est trop grande, elle est réduite. Si elle est moins large que la page, elle est agrandie.

Ce comportement peut être modifié avec l'attribut `fit` qui accepte les valeurs `contain` (par défaut) et `none` (aucune adaptation)

```
:title-page-background-image:  
image:images/image_de_fond_pour_page_de_garde.png[position=top, fit=contain]
```



Figure 141. image de fond sans adaptation à la largeur de la page de garde

Pour préciser une taille spécifique à l'image de fond, il faut annuler l'adaptation avec `fit=None` puis mentionner une largeur d'image. Le mieux est de la préciser en fonction de la largeur du pdf (attribut `pdfwidth`, mais cela peut être fait avec l'attribut `width` et une valeur entière sans unité)

```
:title-page-background-image:  
image:images/image_de_fond_pour_page_de_garde.png[position=top, fit=none,  
pdfwidth=50%] ①
```

① l'image ne doit pas s'adapter à l'espace disponible, c'est pourquoi `fit=none`. L'image doit faire 50% de la largeur du pdf généré.

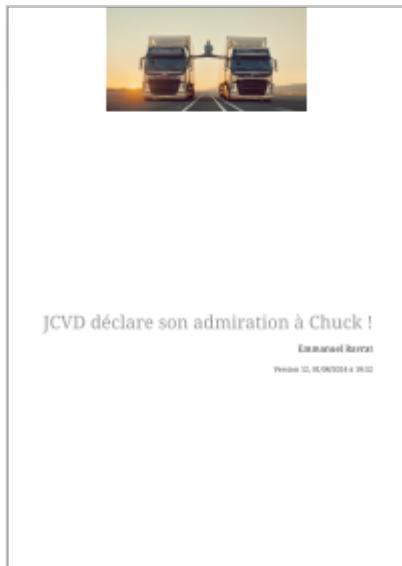


Figure 142. page de garde avec image redimensionnée arbitrairement

Si une image de fond pour la page de garde est définie via le fichier de thème, c'est l'image définie via l'attribut de document `title-page-background-image` qui est prioritaire.

Cela est logique car une image définie dans un fichier de thème est utilisée par tous les documents qui partage ce thème. L'attribut `title-page-background-image` permet de modifier ce comportement pour un document en particulier.

### 29.2.3. Mise en forme des éléments textuels de la page de garde avec un fichier de thème

Si vous avez un besoin trop complexe pour être couvert par les possibilités abordées dans cette partie, vous trouverez une solution dans la [partie 29.3, “Besoin d'une page de garde complexe”](#).

La mise en forme des éléments textuels de la page de garde ou *title-page* en anglais se fait via le fichier de thème. Si vous ne savez pas lier un fichier de thème ou si cette notion n'est pas acquise, le chapitre [chapitre 26, Le fichier de thème PDF](#) est un préalable nécessaire.

Je pars du fichier AsciiDoc suivant :

```
= JCVD déclare son admiration à Chuck !  
:doctype: book
```

```
:pdf-theme: ./my-theme.yml ①
:author: Emmanuel Ravrat
:email: une_adresse@mail.fr
:revdate: 01/08/2024 à 19:52
:revnumber: 12

== Déjà à 5 ans !

== A l'adolescence !

== A l'âge adulte...
```

① Le fichier AsciiDoc utilise le fichier de thème `my-theme.yml` situé dans le même répertoire que lui.

# JCVD déclare son admiration à Chuck !

Emmanuel Ravrat

Version 12, 01/08/2024 à 19:52

Figure 143. rendu de la page de garde avec le thème par défaut

Le fichier de thème `my-theme.yml` doit avoir au minimum le contenu suivant :

```
extends: default ①
```

① permet de reprendre toute la configuration du thème par défaut sans quoi il faudrait absolument tout configurer !

Sur la page de garde actuelle, nous avons les éléments suivants :

- le titre du document
- l'auteur
- les informations de révision
- la page en elle-même

Il peut y avoir en plus :

- une image de fond
- un logo

Voici le contenu du fichier de thème. La mise en page et en forme de la page de garde se fait sous la clé `title-page`. Toutes les explications sont en commentaires.

```
extends: default #reprend la configuration du thème par défaut
title-page: #contient la configuration relative à la page de garde
    background-color: fff4e6 #couleur de fond de la page de garde
    background-image: image:images/vandamme_et_chuck.png[position=bottom,
pdfwidth=70%, fit=none] #image de fond de la page de garde
```

```
font-color: #094ce8 #bleu / n'impacte que la ligne avec le numéro de version
text-align: left # aligne le contenu textuel à gauche
logo: #configuration relative au logo placé sur la page de garde
    image: image:images/logo_chuck.png[width=200] #image du logo placé sur la page
de garde
    top: 25% # 25% de la hauteur du contenu, si 25vh, alors c'est 25% de la page
entière
title: # titre du document
    font-color: c00c00 #rouge chuck !
    font-size: 4rem
    top: 0 # par défaut, c'est 40%
authors: #configuration relative à la ligne des auteurs
    font-color: 2f9407 #vert
    font-style: bold_italic # texte en gras et en italique
    font-size: 1.5rem
revision: #configuration relative à la ligne qui contient le numéro de version
(écrase la configuration des
    font-color: 123eb0 #bleu
    font-size: 2rem
```



Figure 144. rendu de la personnalisation de la page de garde via un fichier de thème



Si une image de logo et/ou une image de fond sont définies respectivement via les attributs de document `title-logo-image` et `title_page_background-image`, elles auront priorité sur les images définies dans le fichier de thème.

Si vous voulez utiliser d'autres propriétés de mise en forme, voici les liens utiles vers la documentation :

- clés pour contrôler la page de garde
- clés pour contrôler le logo
- clés pour mettre en page le titre du document
- clés pour mettre en page les auteurs

-  clés pour mettre en page les informations de révision



L'intérêt d'utiliser un fichier de thème pour la mise en page de la page de garde est sa réutilisation par d'autres documents AsciiDoc.



Pour partager facilement un fichier de thème entre plusieurs documents AsciiDoc, je vous conseille de lire le chapitre  chapitre 26, *Le fichier de thème PDF* et plus particulièrement la  partie 26.2.2, “Un seul fichier de thème pour tous les fichiers asciidoc !”.

## 29.3. Besoin d'une page de garde complexe

Si vous avez une page de garde avec plusieurs images, des éléments textuels différents de ceux proposés par défaut, etc., vous pouvez utiliser comme page de garde une page PDF ou une image !

Admettons que je veuille ajouter la page de garde suivante :



Figure 145. page de garde "complexe"

Il n'est pas possible de faire cela avec AsciiDoc (ou du moins pas sans faire du bricolage et se faire un bon mal de tête).

Par contre, c'est très facile d'utiliser l'image comme image de couverture.

Dans le fichier de thème, il faut ajouter la clé `cover` pour la gestion de la couverture. C'est une clé de premier niveau, c'est-à-dire qu'elle est collée contre la marge de gauche :

fichier `my-theme.yml`

```
extends: default #reprend la configuration du thème par défaut
title-page: #contient la configuration relative à la page de garde
    background-color: fff4e6 #couleur de fond de la page de garde
    #ici la configuration des clés "title", "authors", etc.
cover: ①
```

```
front: ②
image: image:images/cover_1.png[] ③
```

- ① clé indiquant la configuration de la couverture
- ② clé indiquant la configuration de couverture de devant
- ③ image à utiliser pour la couverture de devant. Celle-ci se situe dans le dossier image situé au même niveau que le fichier de thème. Prenez n'importe quelle image pour faire le test avec l'attribut `fit=fill` pour remplir toute la page.



Je recommande fortement de créer une image en 21 x 29.7cm afin qu'elle ne soit pas redimensionnée automatiquement. Cela évite d'avoir à "jouer" avec les attributs `fit=fill` (ou une autre valeur), `position`, etc. (voir la partie qui traite de l'ajout d'une image de fond)

Générez votre fichier pdf à l'aide du bouton [ PDF ].

Vous devriez avoir le résultat suivant :

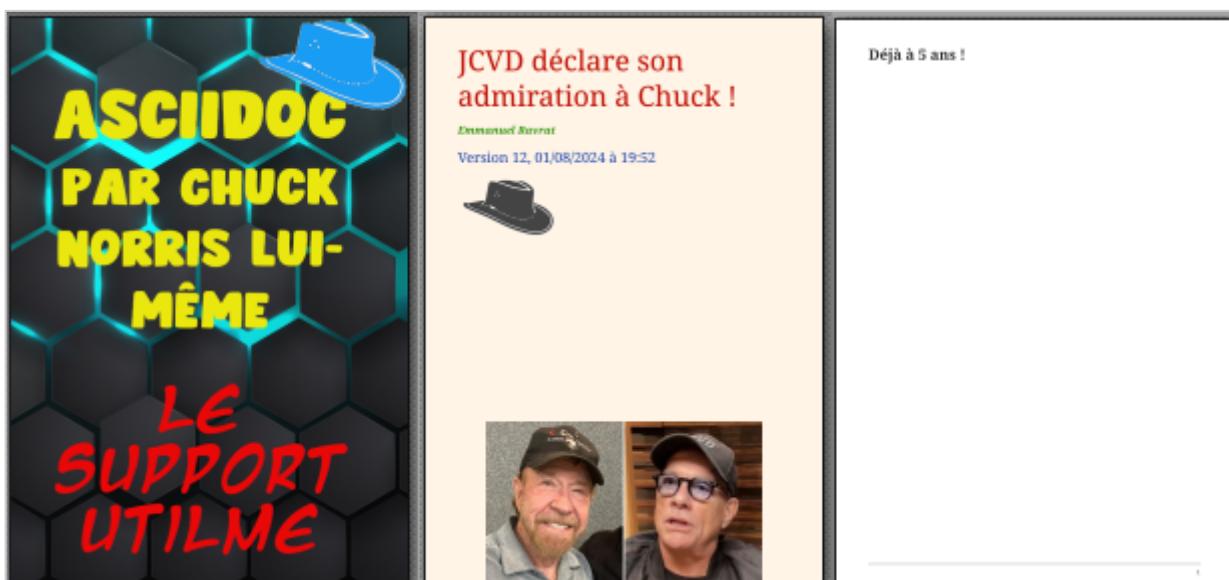


Figure 146. rendu des 3 premières pages du pdf



Comme vous pouvez le constater, la page de garde est toujours présente.

Il est possible de désactiver la page de garde au niveau d'un document en déclarant l'attribut de document `notitle` sous le titre principal :

```
= JCVD déclare son admiration à Chuck !
:doctype: book
:notitle: ①
```

- ① désactive la page de garde. Elle ne sera pas rendue dans le fichier pdf généré.



A ce jour, le 01/08/2024, le plugin AsciiDoc ne prend pas en charge l'attribut `notitle`.

J'ai toutefois testé la génération du pdf en ligne de commande et cela a bien fonctionné. Il faut donc attendre une mise à jour du plugin qui corrige ce bogue.

Sachez qu'il est possible de désactiver la page de titre via le fichier de thème en affectant la valeur `false` à la clé `title-page` tel que :

*fichier de thème*

```
extends: default
title-page: false ①
```

① désactive la génération de la page de garde pour tous les fichiers qui utilisent le fichier de thème. Il ne doit plus y avoir de sous-clés sous la clé `title-page`.

Pour utiliser une couverture différente pour chaque document AsciiDoc, l'astuce consiste à stocker le nom de l'image de couverture dans un attribut et de le déclarer dans l'entête du document.

Votre document AsciiDoc :

```
= JCVD déclare son admiration à Chuck !
:doctype: book
:notitle: ①
:pdf-theme: ./my-theme.yml
:cover_front_image: cover_asciichick.png ②
//... autres attributs d'entête de document
```

① désactivation de la page de garde

② l'attribut `cover_front_image` contient le nom du fichier d'image à utiliser pour le fichier AsciiDoc courant (ce pourrait être un chemin relatif ou absolu). Le nom de l'attribut est totalement libre. Il doit absolument être déclaré avec les attributs d'entête, c'est-à-dire immédiatement sous le titre

Ensuite, l'attribut `cover_front_image` est référencé depuis le fichier de thème :

```
extends: default
title-page: false ①
cover:
  front:
    image: image:images/{cover_front_image}[] ②
```

① si vous souhaitez désactiver la page de garde pour tous les fichiers qui utilisent le thème

② l'attribut `cover_front_image` sera remplacé par sa valeur lors de la génération du fichier pdf.

Si la page que vous voulez utiliser comme page de garde figure dans une page d'un fichier pdf, il vous suffit de remplacer le nom de l'image par le nom du fichier pdf (à voir si vous conserver le chemin qui pointe le dossier des images) :

```
= JCVD déclare son admiration à Chuck !
:doctype: book
:notitle:
:pdf-theme: ./my-theme.yml
:cover_front_image: un_fichier_pdf.pdf ①
//... autres attributs d'entête de document
```

① il est indiqué un fichier pdf même s'il fait plusieurs pages

Dans le fichier de thème, cela ne change rien en dehors du fait que l'on peut spécifier le numéro de page du pdf à utiliser :

```
extends: default
title-page: false
cover:
  front:
    image: image:images/{cover_front_image}[page=4] ①
```

① l'attribut `cover_front_image` sera remplacé par sa valeur lors de la génération du fichier pdf. Ici, ce sera la page 4 du fichier pdf. Si aucun numéro de page n'est spécifié, c'est la première page qui est utilisée.

Maintenant que vous savez gérer une page de garde (en fait c'est une image de couverture que l'on utilise comme telle), autant vous parler de la possibilité d'ajouter une page de couverture arrière.

Tout ce que j'ai écrit pour la couverture avant s'applique à la couverture arrière. Il n'y a que la clé dans le fichier de thème qui change :

```
extends: default
title-page: false
cover:
  front:
    image: image:images/{cover_front_image}[]
  back: ①
    image: image:images/{cover_back_image}[] ②
```

① configuration de la couverture arrière

② image utilisée pour la couverture arrière

Fichier AsciiDoc :

```
= JCVD déclare son admiration à Chuck !
:doctype: book
:notitle:
```

```
:pdf-theme: ./my-theme.yml
//...autres attributs d'entête
:cover_front_image: cover_asciichick.png
:cover_back_image: cover_1.png ①
```

① attribut contenant le nom d'image de la couverture arrière. Cet attribut est référencé dans le fichier de thème

Vous pouvez bien sûr utiliser l'astuce qui consiste à stocker le nom du fichier image (ou pdf) dans un attribut déclaré au niveau de l'entête du document.

Après avoir généré votre fichier pdf, vous avez un joli support avec une belle couverture avant et arrière !

Vous pouvez aller encore plus loin avec [la page de la documentation dédiée aux couvertures](#).

# Index

## A

alignement horizontal  
  cellule, 99  
alterner entre ligne ombrée et non ombrée, 116  
ancre, 59  
ancrer les titres, 60  
annexes d'un document, 139  
Asciidoctor PDF, 175  
attribut, 24  
  activer, 125, 150  
  affectation, 126  
  afficher la valeur, 127  
  création, 125  
  défini, 150  
  désactiver, 135  
  indéfini, 152  
  tester sa définition, 150  
attribut de document, 25, 25, 27, 71  
attribut indéfini, 152  
attributs de document, 124  
attributs de l'utilisateur, 131  
attributs intégrés, 124, 129  
auteur, 188  
  plusieurs auteurs, 189

## B

bang, 153  
bloc de liste AsciiDoc, 73  
bloc d'avertissement, 68  
bloc littéral, 31  
bordure d'un tableau, 113

## C

callouts, 78  
caractère de continuation de ligne, 126  
cellule de tableau, 91  
centrer un tableau, 105  
checklist, 45  
chemin absolu, 55  
code source, 74  
commentaire AsciiDoc, 18  
commentaire multiligne, 18  
commentaire sur une ligne, 18  
compteurs, 137  
couverture

arrière, 207  
avant, 204  
csv, 117

## D

date de révision, 188  
dimension des colonnes d'un tableau, 93  
directive ifeval, 155  
doctype, 144  
décalage de prévisualisation, 160  
défini, 125

## E

entête de page, 189  
entêtes de ligne, 107

## F

fichier de thème, 175, 189

## I

icônes d'avertissement, 71  
icône  
  ajouter un rôle, 57  
  insertion, 56  
  modifier la taille, 57  
ifdef, 150  
ifeval, 155  
image  
  alignement, 50  
  remplacer une image existante, 49  
  supprimer une image, 50  
image de fond, 199  
image page de garde  
  adapter la largeur, 200  
  positionner en haut, 199  
inclure des sous fichiers, 162  
inclusion d'un fichier avec images, 171  
indentation du code, 77  
index, 146

## L

lien avec espaces vers un fichier, 55  
lien avec texte personnalisé, 52  
lien désactivé, 53  
lien hypertexte automatique, 52

lien relatif vers un fichier, 54  
ligne d'un tableau, 91  
liste avec une image, 41  
liste imbriquée, 40  
liste non ordonnée, 38, 38  
liste non ordonnée imbriquée dans une liste ordonnée, 45  
Liste numérotée, 42  
liste ordonnée, 41  
Liste ordonnée avec des lettres, 43  
liste ordonnée imbriquée dans une liste ordonnée, 44  
légende d'une image  
    alignement, 51  
l'entête de document asciidoc, 25

## M

macro de passage en ligne, 127  
mode paysage, 121  
mode portrait, 121  
multiplicateur de colonne, 93  
métadonnée  
    auteur, 188  
    date de révision, 188  
métadonnées, 188  
    numéro de version, 188

## N

numéro de version, 188  
    label, 189  
numéro d'une liste ordonnée, 44  
numéros de légende, 78  
Numéroter les titres, 27

## O

opérateur bang, 154  
opérateur de style, 106  
opérateur d'alignement dans AsciiDoc, 99

## P

page de garde, 144, 188, 195  
    besoin complexe, 204  
    complexe et différente par document, 206  
    désactiver depuis le thème, 206  
    désactiver via un attribut, 205  
    forcer sa génération, 196  
    mise en forme, 201  
pagination, 193

désactivation, 194  
paragraphe, 29  
paragraphe de préambule, 32  
personnalisation de l'étiquette de titre, 47  
pied de page, 193  
principe DRY, 169  
profondeur de la table des matières, 27  
profondeur des titres à numéroter, 28  
préambule document AsciiDoc, 32

## R

retour à la ligne, 30  
référence croisée, 59  
région de code taguée, 84  
rôle AsciiDoc, 35  
rôle intégré, 35  
rôle personnalisé, 183

## S

sous-titre, 20

## T

table des matières, 20  
titre, 20, 20  
    modifier le niveau globalement, 165  
    modifier le niveau par fichier, 164  
titre de la table des matières, 28  
type de document, 144

## V

variable, 124