

# Rédaction de documents techniques en BTS SN

## Sommaire

1. Objectifs .....	1
2. Présentation .....	1
2.1. Les outils .....	1
2.2. Langage de description de format de document .....	2
2.3. Le fond et la forme .....	2
2.4. Quelques formats de document .....	3
2.5. Spécificités des documents techniques en informatique .....	6
2.6. Les formats simples .....	7
3. Les formats utilisés par les développeurs .....	8
3.1. Markdown .....	8
3.1.1. Pandoc .....	9
3.1.2. Édition avec aperçu .....	14
3.2. AsciiDoc .....	15
3.2.1. AsciiDoctor .....	16
3.2.2. Édition avec aperçu .....	20
4. Voir aussi .....	21

Thierry Vaira - <[tvaira@free.fr](mailto:tvaira@free.fr)> - version v0.3 - 23/08/2021 - [tvaira.free.fr](http://tvaira.free.fr)

## 1. Objectifs

Dans le cadre du développement logiciel, il est habituel de produire des documents techniques (documentations, manuels, tutoriels, compte-rendus, ...).

Quel outil utilisé ? Quel format ? etc ... À la recherche de la solution simple et rapide pour produire des documents de qualité !

## 2. Présentation

### 2.1. Les outils

De manière générale, il existe deux types d'outils pour produire ce type de documents :

- le **traitement de texte** qui consiste à mettre en forme un texte d'un point de vue typographique afin de produire un **document**. Celui-ci possède un format de fichier contenant notamment des

informations sur la structuration et la présentation. Ce type de logiciels (comme **Word** de Microsoft ou **Writer** de LibreOffice) sont souvent WYSIWYG.



« **WYSIWYG** » est l'acronyme de « *what you see is what you get* », signifiant littéralement en français « ce que vous voyez est ce que vous obtenez ». Un logiciel WYSIWYG est donc un logiciel qui dispose d'une interface graphique qui permet à l'utilisateur d'éditer son document tel qu'il sera publié.

- **l'éditeur de texte** qui permet de saisir et modifier interactivement des **fichiers textes** (codés en ASCII ou Unicode) sans mise en page, ni mise en forme car cela ne le concerne généralement pas.



Un éditeur de texte est un outil incontournable pour certaines tâches informatiques de base comme l'administration de système et le développement de logiciels.

## 2.2. Langage de description de format de document

Un langage de description de format de document est un langage permettant de définir un jeu de règles et contraintes qui seront utilisées pour savoir si une instance de document est valide par rapport à ce même jeu de règles et contraintes.

Pour les documents ou fichiers texte, on utilisera principalement des langages balisés : **SGML** (*Standard Generalized Markup Language*), **XML** (*Extensible Markup Language*), **HTML** (*Hypertext Markup Language*).

Parmi les autres langages de description de texte, on peut citer:

- **LaTeX** : notamment pour les textes scientifiques comprenant des formules mathématiques
- **roff** : historiquement, c'était un langage de description de texte utilisé autrefois sous Unix. Il sert en particulier au formatage des pages de manuel (**man**). Lien : [Un tutoriel pour créer des pages man](#).



Il existe aussi des formats dédiés à la représentation de **données**, comme par exemple : **JSON** (*JavaScript Object Notation*) et **YAML** (*Yet Another Markup Language*).

## 2.3. Le fond et la forme

En informatique, la séparation du fond et de la forme est un point important de la création et de la gestion d'un document par un outil informatique.

La séparation du fond et de la forme consiste à séparer le message d'un document (et sa structure) de sa présentation.



La séparation du fond et de la forme n'est pas une nécessité en soi ; elle n'est pas liée à une contrainte du logiciel ou du matériel. Mais elle améliore la qualité des documents car l'auteur se concentre tout d'abord sur le contenu du document. La mise en forme est traitée après, ou parfois avant, mais dans tous les cas séparément.

Inconvénients des logiciels de traitement de texte WYSIWYG :

- les documents écrits avec des logiciels WYSIWYG sont très sensibles aux changements de format. Il est donc nécessaire dans ce cas de refaire toute la mise en forme pour chaque support différent.
- les logiciels « WYSIWYG » sont souvent difficiles à utiliser pour réaliser des mises en pages précises et complexes (typographie fine, équations, langues non latines). De plus, la restriction au visuel peut être source d'erreurs, car on peut avoir une apparence correcte alors que la structure n'est pas correctement indiquée.
- les traitements de texte type WYSIWYG permettent (sans l'obliger) la séparation du fond et de la forme mais sont le plus souvent mal utilisés de manière spontanée et par manque de formation.



Avec un traitement de texte, la séparation du fond et de la forme exigerait de ne placer aucun caractère de mise en page superflu : paragraphes vides, tabulations en début de ligne (ou d'ailleurs où que ce soit), sauts de page non exigés par la fonction, espaces multiples, etc. En effet, tout caractère de mise en page non indispensable est de la « forme » qui pollue le « fond ».

D'autre part, ce n'est pas un texte en rouge de taille 13 qui définit que c'est un titre. Il faut d'abord définir que c'est un titre (le fond) puis on appliquera la couleur rouge et la taille 13 (la forme) à cet élément de structure (le titre).

Un même document peut être multisupport. Le multisupport est la caractéristique d'un document à être présenté sur différents supports, par exemple : un support papier, un diaporama, un livre électronique et/ou un site web interactif. Chacun de ces supports aura des exigences de présentation différentes.

## 2.4. Quelques formats de document

Les technologies ci-dessous assurent la séparation du fond et de la forme :

- [LaTeX](#) :

**LaTeX** est un langage et un système de composition de documents. Il s'agit d'une collection de macro-commandes destinées à faciliter l'utilisation du « processeur de texte » [TeX](#) de [Donald Knuth](#).

LaTeX permet de rédiger des documents dont la mise en page est réalisée automatiquement en se conformant du mieux possible à des normes typographiques. Une fonctionnalité distinctive de LaTeX est son mode mathématique, qui permet de composer des formules complexes.

LaTeX est particulièrement utilisé dans les domaines techniques et scientifiques pour la production de documents de taille moyenne (tels que des articles) ou importante (thèses ou livres, par

exemple).

Le langage et système de composition de documents LaTeX dispose d'environnements ou d'instructions permettant la séparation du fond et de la forme :

*Exemple pour un document LaTeX*

```
\author{Thierry Vaira}  
\title{Un Exemple}  
\begin{document}  
\section{D\'ebut}  
A faire\dots  
\section{Suite et fin}  
On verra plus tard.  
\end{document}
```



Beamer est une classe LaTeX adaptée à la création de présentations (des diaporamas). Elle possède une syntaxe spéciale pour définir les pages (« transparents », « diapositives »), appelées *frames*.

- [DocBook](#) :

**DocBook** est un langage de balisage sémantique pour la documentation technique. À l'origine prévu pour écrire de la documentation technique portée sur le domaine informatique (matériel et logiciel), il peut être utilisé pour n'importe quel type de documentation.

En tant que langage sémantique DocBook permet à ses utilisateurs de créer du contenu sous une forme neutre vis-à-vis de la présentation qui ne fait que capturer la structure logique du contenu; contenu qui peut ensuite être publié dans une grande variété de formats, notamment HTML, XHTML, EPUB, PDF, pages de man, sans obliger les utilisateurs à faire des changements dans le contenu source. En d'autres termes, quand un document est écrit dans le format DocBook il devient facilement portable vers d'autres formats.

Aujourd'hui, DocBook respecte le standard XML et sa conversion vers un autre format se fait par l'intermédiaire de feuilles de style XSL.

*Exemple pour un article en Docbook version 4*

```
<book id="exemple_de_livre">  
  <title>Livre très simple</title>  
  <chapter id="exemple_de_chapitre">  
    <title>Chapitre très court</title>  
    <para>Bonjour tout le monde !</para>  
    <para>Ceci est un autre paragraphe...</para>  
  </chapter>  
</book>
```

- [HTML](#) / [CSS](#) :

**HTML** (*HyperText Markup Language*) est un langage de balisage conçu pour représenter les pages web. Ce langage permet notamment de structurer sémantiquement la page mais aussi de mettre en forme le contenu. Il est souvent utilisé conjointement avec des feuilles de style en cascade (CSS). Ces feuilles de style CSS (*Cascading Style Sheets*) forment un langage informatique qui décrit la présentation des documents HTML.

Les documents web sont donc généralement construits afin d'assurer une séparation du fond et de la forme : le contenu et sa structure dans un fichier HTML et sa mise en forme dans une feuille de style CSS.

*Exemple pour une simple page web en HTML*

```
<html>
  <head>
    <title>Le titre du document</title>
  </head>
  <body>
    <h1>Un titre de niveau 1</h1>
    <p>Un paragraphe</p>
  </body>
</html>
```

*Exemple pour une simple feuille de style CSS*

```
body {
  background-color: green; /* Couleur de fond */
}
p {
  color: red; /* Couleur du texte */
  font-size: 20px; /* Taille du texte en px (pixel) */
}
```

- [OpenDocument](#) :

**OpenDocument** est un format ouvert (et normalisé) de données pour les applications bureautiques : traitements de texte, tableurs, présentations, diagrammes, dessins et base de données bureautique. De nombreux logiciels utilisent cette norme, principalement OpenOffice.org, **LibreOffice** (qui dérive d'OpenOffice.org), NeoOffice, StarOffice, KOffice.

Le format OpenDocument regroupe différents les formats (avec les extensions de documents associées) : texte formaté **.odt** , tableur **.ods**, etc ...

Techniquement, un fichier OpenDocument est une archive compressée (PKZIP) regroupant un ensemble de fichiers XML et de répertoires. Le format OpenDocument soutient une forte séparation entre contenu, mise en page et métadonnées. Le fichier **content.xml** contient le contenu réel du document (excepté le contenu binaire telles les images qui sont stockées dans des fichiers séparés).



OpenDocument représente le premier effort de normalisation des formats de fichier de la bureautique. **En France, le format OpenDocument est le seul format recommandé comme format bureautique par le référentiel général d'interopérabilité depuis sa version 2.0 validé le 20 avril 2016.**

- [Office Open XML](#) :

**Office Open XML** est une norme ISO/CEI 29500 créée par Microsoft, destinée à répondre à la demande d'interopérabilité dans les environnements de bureautique et à concurrencer la solution d'interopérabilité OpenDocument soutenue par tous les autres éditeurs de suites bureautiques.

Ce format est à l'instar du format OpenDocument structuré en XML et Zip.

Inconvénients de ces types de technologies :

- syntaxe riche et complexe (notamment LaTeX)
- nécessite souvent un logiciel spécifique (comme un traitement de texte) pour manipuler ces formats
- contenu du document est difficilement lisible sans une transformation dans un format de publication ou un système d'aperçu (*preview*)
- format parfois spécifique à un domaine
- nécessite un certain temps d'apprentissage et ensuite de rédaction

## 2.5. Spécificités des documents techniques en informatique

Les documents techniques en informatique contiennent des éléments spécifiques qui sont :

- des fragments de code source dans un langage de programmation
- des commandes
- des noms de programme, des entrées au clavier et des messages à l'écran

Ces éléments spécifiques répondent à quelques règles typographiques que tout technicien en Informatique devrait respecter :

- utilisation d'une police (fonte de caractères) à chasse fixe (type **Courier**)
- utilisation déconseillée (interdite ?) des images (capture d'écran par exemple) pour représenter un contenu texte
- utilisation si possible d'une coloration syntaxique (les mots clés du langage posséderont une couleur unique et distincte)

Courier a été dessinée par Howard Kettler en 1955, pour le compte d'IBM qui l'utilisera pour la première fois en 1961 sur sa machine à écrire Selectric. C'est une police à chasse fixe (tous les caractères ont exactement la même largeur). La police **Liberation Mono** est une alternative libre qui possède une métrique équivalente.

Pour respecter la séparation du fond et de la forme, il est nécessaire de structurer le document en indiquant ces éléments spécifiques et de leur appliquer ensuite un style adapté.

## 2.6. Les formats simples

Ces formats ont pour objectif d'offrir une syntaxe facile à lire et à écrire tout en assurant une structure du document.

Un langage de balisage léger est un type de langage de balisage utilisant une syntaxe simple, conçu pour être aisé à saisir avec un **éditeur de texte** simple, et facile à lire dans sa forme non formatée.

Exemple de formats très utilisés par les développeurs :

- [Markdown](#)
- [AsciiDoc](#)



Les fichiers texte au format Markdown ou AsciiDoc permettent aussi d'être versionnés dans un logiciel de gestion de versions (comme [git](#), [subversion](#), ...) et donc utilisables dans le cadre d'un travail collaboratif.

Les langages suivants servent aussi à produire des documents : atx, BBCode, Creole, Epytext, Haml, JsonML, MakeDoc, Org-mode, POD, RD, ReStructuredText, Setext, SISU, SPIP, Taxy!, Textile, Txt2tags, UDO, Wikitexte et Xupl.

- Un exemple : [MediaWiki](#) ([Wikitexte](#))

**MediaWiki** est un moteur de wiki pour le Web. Il est utilisé par l'ensemble des projets de la Wikimedia Foundation. Un **wiki** est une application web qui permet la création, la modification et l'illustration collaboratives de pages à l'intérieur d'un site web. Il utilise un langage de balisage et son contenu est modifiable au moyen d'un navigateur web. Un wiki fonctionne grâce à un moteur de wiki : c'est un logiciel installé sur le système hôte du site web. Le **wikitexte** ou **wikicode** est un **langage de balisage léger** qui définit la mise en forme de saisies de contenu utilisateur. Il est le plus souvent utilisé pour écrire les pages d'un wiki. La syntaxe de chaque wiki dépend de l'analyseur syntaxique utilisé.



Le mot « wikiwiki » signifie, en hawaïen, rapide, vite ou informel. Il a été choisi par Ward Cunningham lorsqu'il créa le premier wiki, qu'il appela WikiWikiWeb.

```
= Nouveau chapitre =  
== Nouvelle section ==  
=== Nouvelle sous-section ===  
==== Nouveau paragraphe ====  
  
* Les listes sont bien pratiques :  
** Elles permettent d'organiser les données  
** Elles embellissent le document  
  
Lien externe : [http://fr.wikipedia.org Wikipédia]
```

## 3. Les formats utilisés par les développeurs

### 3.1. Markdown

**Markdown** est un langage de balisage léger créé par John Gruber en 2004. Son but est d'offrir une syntaxe facile à lire et à écrire. Un simple **éditeur de texte** suffit pour créer des documents en Markdown.

*Le logo Markdown*



Un document formaté selon Markdown devrait pouvoir être publié comme tel, en texte, sans donner l'impression qu'il a été marqué par des balises ou des instructions de formatage.

Un document rédigé en Markdown peut être converti facilement en HTML, PDF, ODT, EPUB etc ...

Quelques liens :

- [Site officiel](#)
- [Le format Markdown](#)
- [La syntaxe](#)
- [Markdown avec VSCode](#)



L'extension par défaut des fichiers Markdown est **.md**.



# Un titre de niveau 1

## Un titre de niveau 2

Un mot en *\*gras\** et un mot en *\_italique\_* et le nom d'une célèbre commande ``ls``.

Extrait d'un code source en C++ :

```
```cpp
int a = 0;
```
```

Une commande sous *\*GNU/Linux\** :

```
```
$ uname --operating-system
GNU/Linux
```
```

Une simple liste de langages :

- C
- C++
- Python

Une liste de tâches :

- [x] déclarer la classe ``Point``
- [] définir la classe ``Point``
- [] tester et valider la classe ``Point``



Un [exemple de document contenant du code C++ rédigé en Markdown](#) et converti en HTML → [exemple-markdown.html](#)

### 3.1.1. Pandoc

Il est possible d'utiliser l'outil **pandoc** pour convertir un fichier au format Markdown dans de nombreux autres formats (HTML, PDF, ODT, EPUB, AsciiDoc, etc ...) et inversement.



Pandoc est considéré comme le couteau suisse ultime de la conversion de documents ! Il supporte plus de quarante formats différents.

Sous Ubuntu :

```
$ sudo apt-get install -y pandoc

$ pandoc --version
pandoc 2.5
Compiled with pandoc-types 1.17.5.4, texmath 0.11.2.2, skylighting 0.7.7
Default user data directory: /home/iris/.pandoc
Copyright (C) 2006-2018 John MacFarlane
Web: http://pandoc.org
This is free software; see the source for copying conditions.
There is no warranty, not even for merchantability or fitness
for a particular purpose.

$ pandoc -h
```

Quelques conversions avec **pandoc** :

```
# format odt
$ pandoc --toc -o exemple.odt exemple.md

# format pdf
$ pandoc --toc -o exemple.pdf exemple.md

# format html
$ pandoc -s -S --toc -o exemple.html exemple.md
// avec une feuille de style externe
$ pandoc -s -S --toc -c exemple.css -A signature.html -o exemple.html exemple.md

# format epub
$ pandoc -S exemple.md -o exemple.epub
```



L'option **--self-contained** permet de produire un fichier HTML autonome sans dépendances externes (notamment pour les images).

Sous Windows :

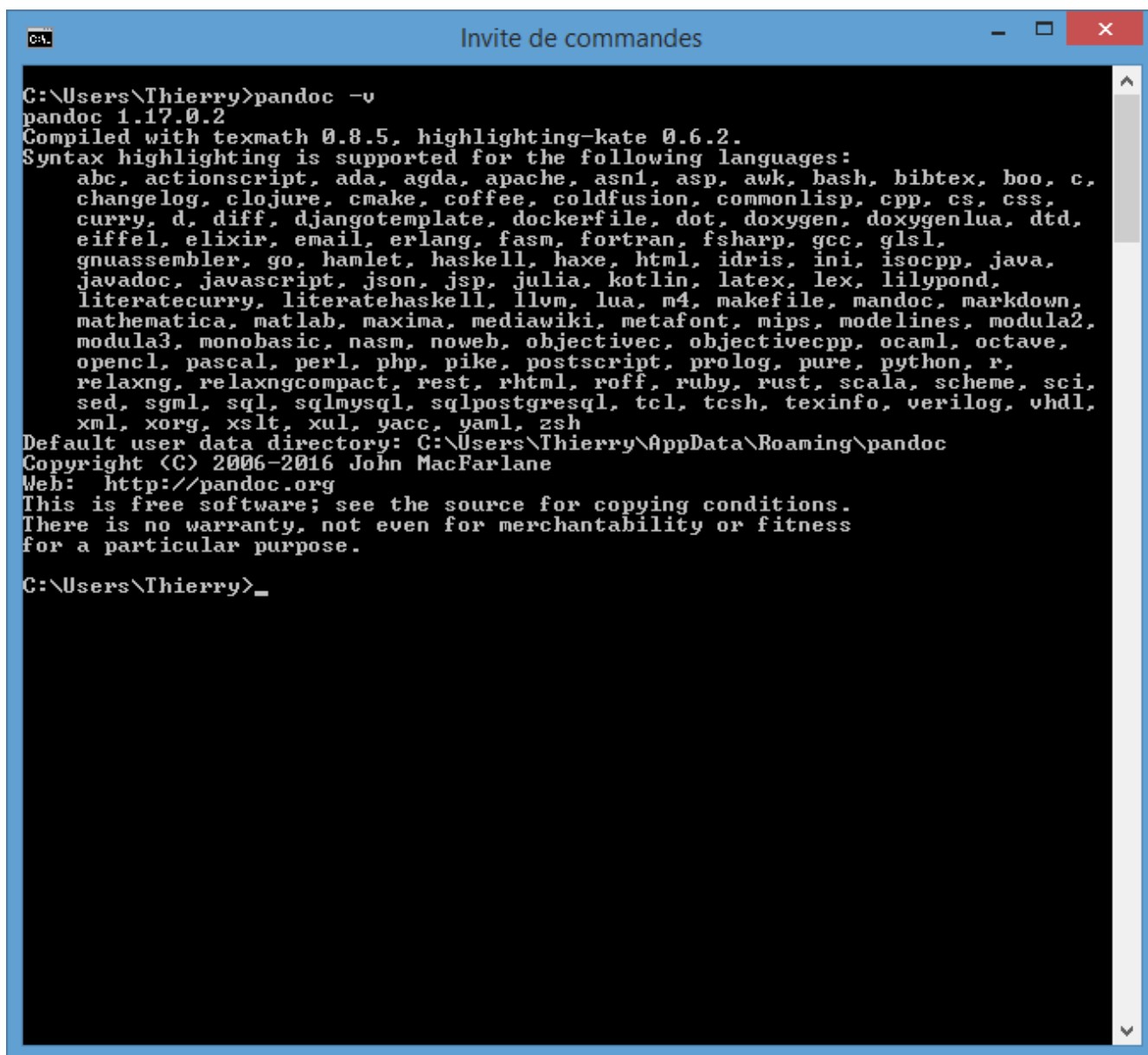
- En ligne de commande (CLI) : [Pandoc](#) et [Package Pandoc msi](#)
- Avec interface graphique (GUI) pour pandoc : [PanConvert](#) et [Package PanConvert msi](#)
- Plugin Word pour pandoc : [Writage](#) et [Package Writge msi](#)

Tests :

- Pandoc :

Installer **pandoc-1.17.0.2-windows.msi**.

Lancer **cmd.exe** puis **pandoc** :



```
C:\Users\Thierry>pandoc -v
pandoc 1.17.0.2
Compiled with texmath 0.8.5, highlighting-kate 0.6.2.
Syntax highlighting is supported for the following languages:
abc, actionscript, ada, agda, apache, asnl, asp, awk, bash, bibtex, boo, c,
changelog, clojure, cmake, coffee, coldfusion, commonlisp, cpp, cs, css,
curry, d, diff, djangotemplate, dockerfile, dot, doxygen, doxygenlua, dtd,
eiffel, elixir, email, erlang, fasm, fortran, fsharp, gcc, glsl,
gnuassembler, go, hamlet, haskell, haxe, html, idris, ini, isocpp, java,
javadoc, javascript, json, jsp, julia, kotlin, latex, lex, lilypond,
literatecurry, literatehaskell, llvm, lua, m4, makefile, mandoc, markdown,
mathematica, matlab, maxima, mediawiki, metafont, mips, modelines, modula2,
modula3, monobasic, nasm, noweb, objectivec, objectivecpp, ocaml, octave,
openc1, pascal, perl, php, pike, postscript, prolog, pure, python, r,
relaxng, relaxngcompact, rest, rhtml, roff, ruby, rust, scala, scheme, sci,
sed, sgml, sql, sqlalchemy, sqlpostgresql, tcl, tcsh, texinfo, verilog, vhdl,
xml, xorg, xslt, xul, yacc, yaml, zsh
Default user data directory: C:\Users\Thierry\AppData\Roaming\pandoc
Copyright (C) 2006-2016 John MacFarlane
Web: http://pandoc.org
This is free software; see the source for copying conditions.
There is no warranty, not even for merchantability or fitness
for a particular purpose.

C:\Users\Thierry>_
```

Voir les options de **pandoc** :

```
C:\Users\Thierry\Documents\markdown-windows>pandoc -h
```

Editer un document **.md** puis le convertir en HTML :

```
C:\Users\Thierry\Documents\markdown-windows>pandoc -s -S --toc -o markdown-
windows.html markdown-windows.md
```

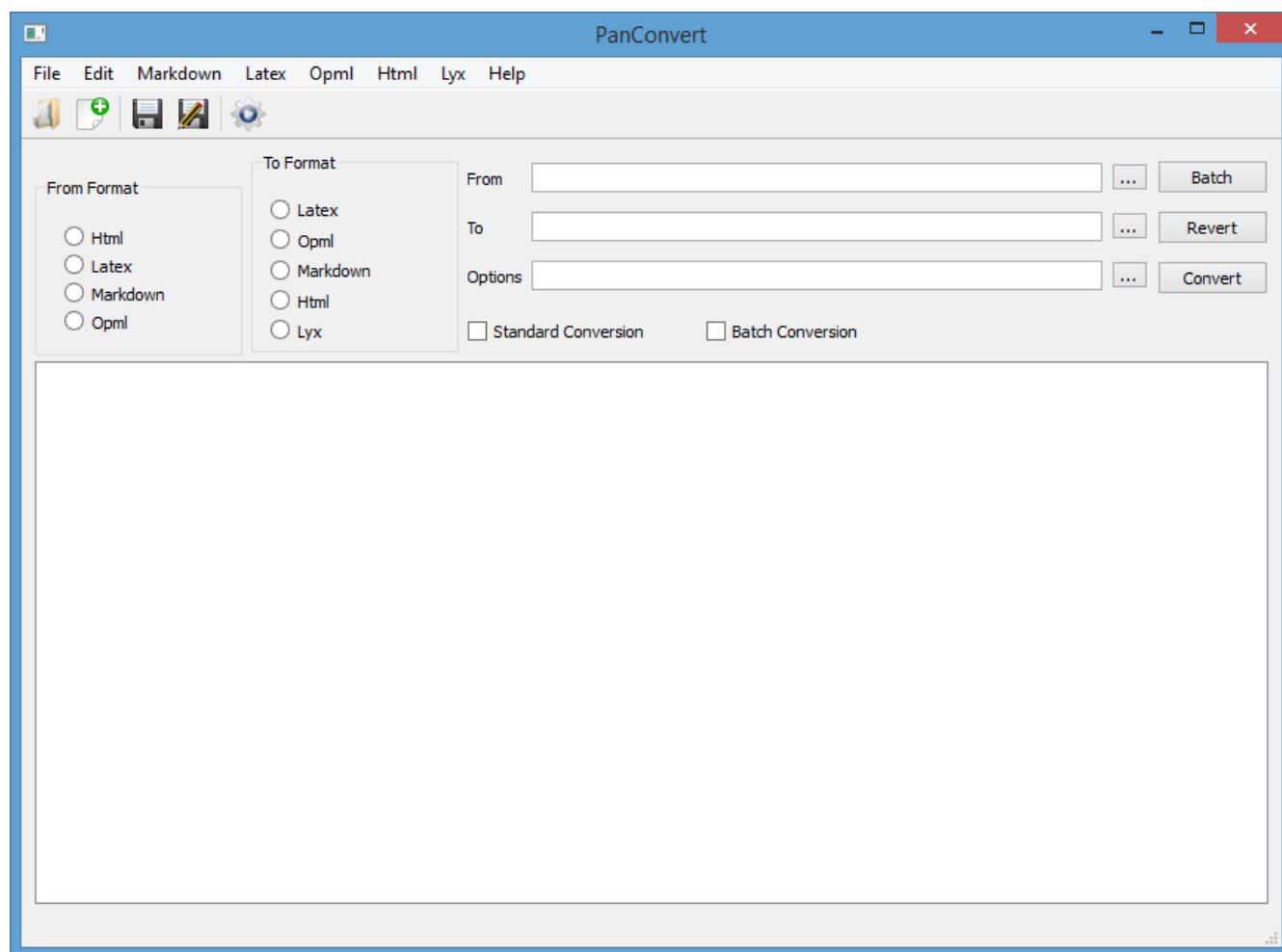
Et le convertir en **.odt** :

```
C:\Users\Thierry\Documents\markdown-windows>pandoc --toc -o markdown-windows.odt
markdown-windows.md
```

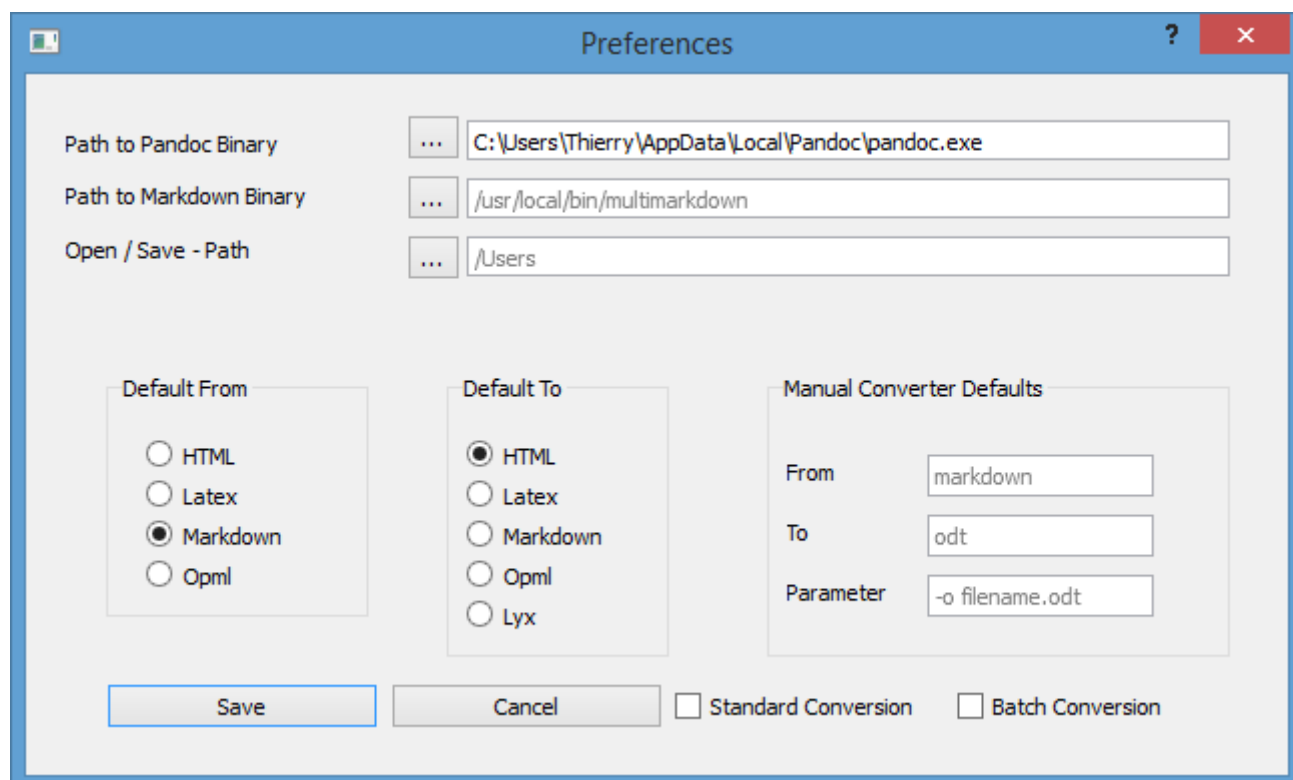
- **PanConvert** :

Installer [PanConvert-0.1.5-win32.msi](#).

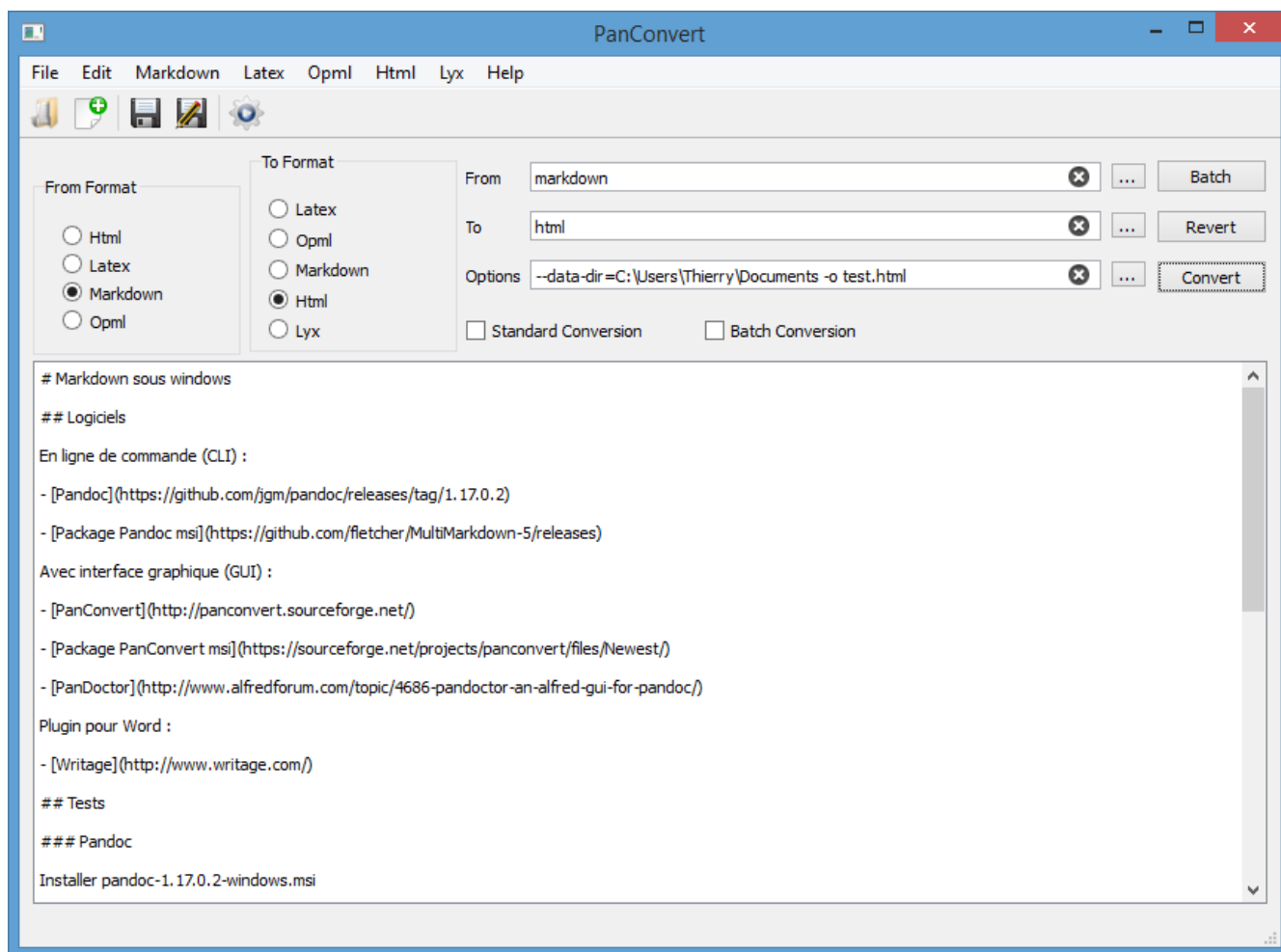
Lancer [Panconvert.exe](#).



Edit → [Préférences](#) pour configurer le chemin vers [pandoc](#) :



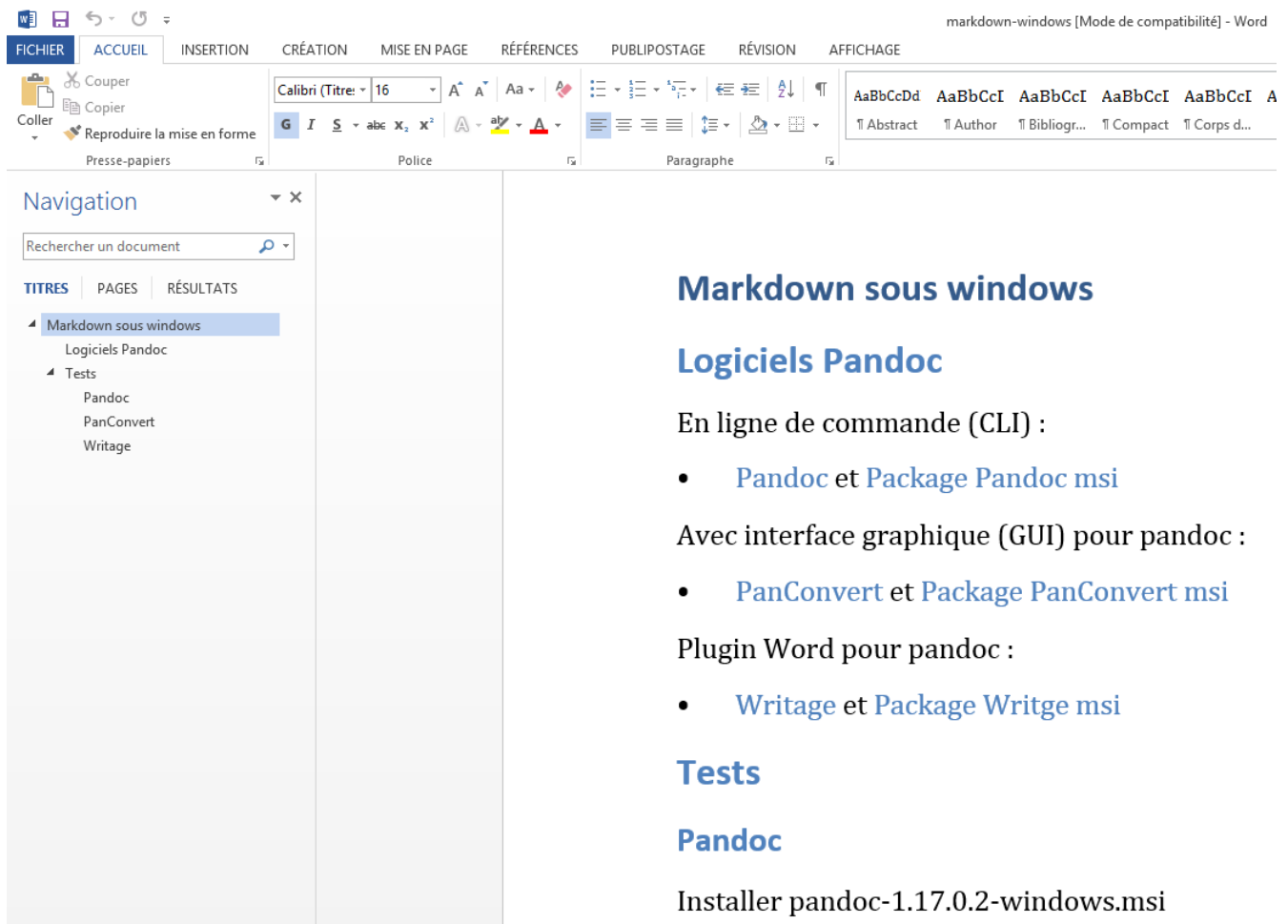
Ouvrir un fichier ou saisir un texte puis le convertir :



- Writage :

Installer **Writage-1.8.3.msi**.

Lancer Word et ouvrir un document markdown :



## Markdown sous windows

### Logiciels Pandoc

En ligne de commande (CLI) :

- [Pandoc](#) et [Package Pandoc msi](#)

Avec interface graphique (GUI) pour pandoc :

- [PanConvert](#) et [Package PanConvert msi](#)

Plugin Word pour pandoc :

- [Writage](#) et [Package Writge msi](#)

### Tests

#### Pandoc

Installer `pandoc-1.17.0.2-windows.msi`

### 3.1.2. Édition avec aperçu

Extension [Markdown Viewer](#) pour les navigateurs Chrome et Firefox.

Il existe de nombreuses extensions Markdown pour [Visual Studio Code](#) :

- [Markdown Docs](#)
- [Markdown Extension Pack](#)
- ...

Il existe des éditeurs en ligne pour Markdown :

- [www.tutorialspoint.com](http://www.tutorialspoint.com)
- [dillinger.io](http://dillinger.io)
- [hackmd.io](http://hackmd.io)
- [demo.hedgedoc.org](http://demo.hedgedoc.org)
- [stackedit.io](http://stackedit.io) (qui est aussi une extension pour Google Drive)
- ...

Markdown est le format des fichiers README pour les projets logiciels hébergés sur [GitHub](#).

Voir aussi : <https://daringfireball.net/projects/markdown/dingus>

## 3.2. AsciiDoc

**AsciiDoc** est un langage de balisage léger proche du langage *Markdown*, proposant une richesse sémantique plus importante. Un simple **éditeur de texte** suffit pour créer des documents en AsciiDoc. Ce document a été rédigé en AsciiDoc !

Syntaxe : <http://asciidoctor.org/docs/asciidoc-syntax-quick-reference/>



### *Markdown vs AsciiDoc*

Comme Markdown, les fichiers **.adoc** (extension par défaut pour des fichiers AsciiDoc) utilisent une syntaxe simple à écrire et à lire. Voici un comparatif entre les deux formats : <http://asciidoctor.org/docs/user-manual/#compared-to-markdown>

### Histoire d'AsciiDoc

AsciiDoc a été publié pour la première fois en novembre 2002 par Stuart Rackham. Il a été conçu dès le départ pour produire des documents professionnels comme DocBook et LaTeX à partir d'une syntaxe abrégée.

*Le logo AsciiDoc*



Lien : <https://asciidoc.org/>

= Un titre de niveau 1

== Un titre de niveau 2

Un mot en *\*gras\** et un mot en *\_italique\_* et le nom d'une célèbre commande ``ls``.

Extrait d'un code source en C++ :

```
[source,cpp]
```

```
----
```

```
int a = 0;
```

```
----
```

Une commande sous *\*GNU/Linux\** :

```
....
```

```
$ uname --operating-system
```

```
GNU/Linux
```

```
....
```

Une simple liste de langages :

- C
- C++
- Python

Une liste de tâches :

- [x] déclarer la classe ``Point``
- [] définir la classe ``Point``
- [] tester et valider la classe ``Point``

Ce document a été rédigé en AsciiDoc ([guide-redaction-btssn.adoc](#)).



Lien : [AsciiDoc cheatsheet](#)

Un [exemple de document contenant du code C++ rédigé en AsciiDoc](#) et converti en HTML → [exemple-asciidoc.html](#)

### 3.2.1. AsciiDoctor

Un programme de conversion (ou suite logicielle) comme [AsciiDoc](#) ou [AsciiDoctor](#) permet de transformer le document source en un format "publiable" (comme XHTML, DocBook ou HTML).

Associé à la suite logicielle AsciiDoc, le programme [a2x](#) permet d'obtenir d'autres formats tels que PDF, TeX, Unix manpages ou Epub. AsciiDoctor intègre des fonctionnalités similaires.





Lien : <http://asciidoctor.org/>

Sous Ubuntu :

```
$ sudo apt-get install -y asciidoctor

$ asciidoctor --version
Asciidoctor 2.0.10 [https://asciidoctor.org]
Runtime Environment (ruby 2.7.0p0 (2019-12-25 revision 647ee6f091) [x86_64-linux-gnu])
(lc:UTF-8 fs:UTF-8 in:UTF-8 ex:UTF-8)

$ sudo gem install asciidoctor-pdf
```



Utiliser `sudo` sous Ubuntu

Une simple conversion en HTML :

```
$ asciidoctor exemple.adoc
```



L'option `-b html5 -a data-uri` permet de produire un fichier HTML autonome sans dépendances externes (notamment pour les images ou fichier inclus). cf. `asciidoctor --help`

```
TARGETDIR_PDF = ../guides-pdf
TARGETDIR_HTML = ../guides-html
PDF := $(patsubst %.adoc,%.pdf,$(wildcard *.adoc))
HTML := $(patsubst %.adoc,%.html,$(wildcard *.adoc))
HTMLOPTIONS=-b html5 -a data-uri
PDFOPTIONS=-r asciidoctor-mathematical -a source-highlighter=pygments -a pygments-style=xcode

$(info Fabrication des guides PDF : $(PDF))
$(info Fabrication des guides HTML : $(HTML))

#all : $(TARGETDIR) $(HTML)
#all : $(TARGETDIR) $(PDF)
all : $(TARGETDIR) $(HTML) $(PDF)

%.html: %.adoc
    asciidoctor $< $(HTMLOPTIONS) -o $(TARGETDIR_HTML)/$@

%.pdf: %.adoc
    asciidoctor-pdf $< $(PDFOPTIONS) -o $(TARGETDIR_PDF)/$@

$(TARGETDIR_PDF):
    mkdir $(TARGETDIR_PDF)

$(TARGETDIR_HTML):
    mkdir $(TARGETDIR_HTML)

.PHONY: clean

clean:
    rm -f $(TARGETDIR_HTML)/*.html $(TARGETDIR_PDF)/*.pdf

rebuild: clean all
```

Asciidoctor ajoute aussi un certain nombre de fonctionnalités qui ne sont pas prises en charge par Asciidoc : <http://asciidoctor.org/docs/asciidoc-asciidoctor-diffs/>



**asciidoctor** ne convertit pas au format PDF. Il faut installer et utiliser **asciidoctor-pdf**

Sous Windows (non testé) :

- télécharger le package [Rubyinstaller](#) pour une version de Windows
- installer [RVM](#) (*Ruby Version Manager*), l'outil en ligne de commande **gem** qui permet d'installer, de gérer et de travailler facilement dans l'environnement Ruby

Ensuite, il faut utiliser la commande `gem` pour installer (ou mettre à jour) `Asciidoctor` dans un terminal :

```
$ gem install asciidoctor
```



Lors de l'utilisation de RVM, les gems sont installées dans un endroit isolé du système.

Installation de gems supplémentaires :

- Pour `:source-highlighter: coderay`

```
$ sudo gem install coderay
```

- Pour `:source-highlighter: pygments`

```
$ sudo gem install pygments.rb  
  
$ pygmentize -L styles
```

Liste de thèmes pour le formatage du code :

```
:highlightjs-theme: xcode  
:highlightjs-theme: androidstudio  
:highlightjs-theme: googlecode  
:highlightjs-theme: github  
:highlightjs-theme: foundation  
:highlightjs-theme: idea  
:highlightjs-theme: rainbow  
:highlightjs-theme: vs  
:highlightjs-theme: sunburst  
:highlightjs-theme: tomorrow  
:highlightjs-theme: railscasts  
:highlightjs-theme: zenburn
```

→ [Expressions mathématiques](#) (`asciimath:[expression]`) : il faut le support `stem` en ajoutant `:stem:`.

- Pour le support `stem` dans les PDF :

```
$ sudo -s  
  
# MATHEMATICAL_SKIP_STRDUP=1 gem install mathematical  
# gem install asciidoctor-mathematical
```



Il faut ajouter l'option `-r asciidoctor-mathematical` dans `asciidoctor-pdf`.

Liste des gems installés :

```
$ sudo gem list

*** LOCAL GEMS ***

addressable (2.7.0)
afm (0.2.2)
Ascii85 (1.0.3)
asciidoctor (2.0.12, 1.5.5)
asciidoctor-mathematical (0.3.5)
asciidoctor-pdf (1.5.3)
asciimath (2.0.2)
...
```

Voir aussi :

*Exemple de tests conditionnels en fonction du format de sortie*

```
ifdef::backend-html5[]
++++
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
++++
:html:
endif::[]

ifdef::backend-pdf[]

endif::[]
```

### 3.2.2. Édition avec aperçu

Extension `Asciidoctor.js` pour les navigateurs Chrome et Firefox.

Extension `AsciiDoc` dans `Visual Studio Code`

Il existe des éditeurs en ligne pour AsciiDoc :

- [www.tutorialspoint.com](http://www.tutorialspoint.com)
- [AsciiDocBox](#)
- [asciidoclive.com](http://asciidoclive.com)

Pour les fichiers hébergés sur GitHub : [gist.asciidoctor.org](https://gist.asciidoctor.org)

Voir aussi :

- <https://asciidoctor.org/docs/editing-asciidoc-with-live-preview/>
- <https://github.com/bodiam/awesome-asciidoc>

## 4. Voir aussi

- Documentation du code avec Doxygen
- Diagrammes UML avec PlantUML
- Conception de pages man

---

Thierry Vaira - <[tvaira@free.fr](mailto:tvaira@free.fr)> - version v0.3 - 23/08/2021 - [tvaira.free.fr](http://tvaira.free.fr)