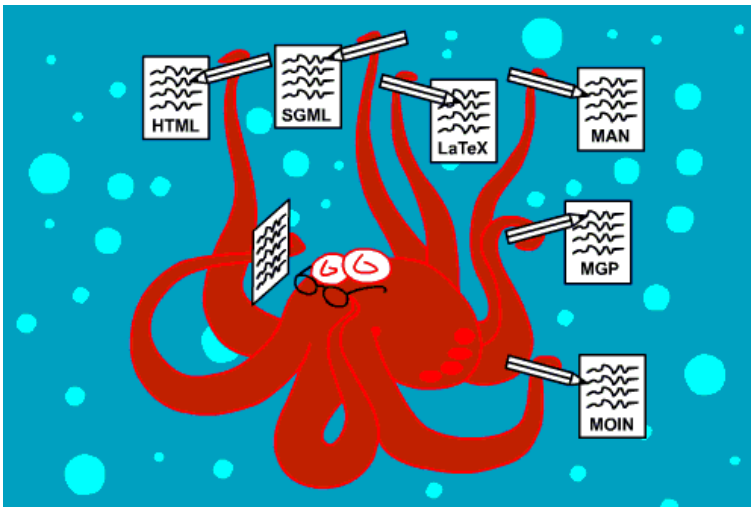
Txt2tags User Guide − French Translation

http://txt2tags.sf.net



Aurélio Marinho Jargas v2.2 (Dec 2004)

Translated by Claude Hiebel (Mar 2005)

Ce guide contient

Ce guide contient

Première partie − Introduction

Les premières questions que vous vous posez

Ce chapitre est un aperçu de txt2tags, qui présente son utilisation et ses caractéristiques.

Quest−ce que c'est ?

Txt2tags est un outil de mise au format de texte et de conversion.

Txt2tags convertit un fichier texte brut avec des petites marques en de multiples formats cibles :

- Document HTML
- Document XHTML
- Document SGML
- Document LaTeX
- Page man UNIX
- Presentation Magic Point
- Page MoinMoin
- DocumentPageMaker 6.0
- Texte brut (sans marques)

Pourquoi dois−je l'utiliser ?

Vous allez trouver txt2tags très pratique si :

- vous avez besoin de publier des documents sous divers formats,
- maintenir des documents à jour sous divers formats,
- écrire des documents techniques ou des manuels,
- vous ne savez pas comment écrire un document dans un format particulier,
- vous n'avez pas un éditeur spécifique pour un certain format,
- vous voulez utiliser un éditeur simple pour faire les mises à jour.

Et la motivation principale est :

- gagner du temps : écrire le **contenu** et ne pas vous soucier de la **mise en forme**.

Pourquoi est−ce un bon choix par rapport à d'autres outils ?

Txt2tags est en très forte croissance, suivant des concepts de base. En voici les points forts :

Fichiers source lisibles	les marques Txt2tags sont très simples, presque naturelles.
Document de sortie lisible	Comme le fichier source, le fichier résultat est lisible, avec des indentations et des lignes courtes.
Des marques compatibles	les marques txt2tags sont suffisamment spécifiques pour aller dans n'importe quel document et ne peuvent pas être confondues avec le contenu.
Des règles simples	Comme les marques, les règles qui s'appliquent sont liées les unes aux autres. Il n'y a pas de "cas spéciaux" ni "d'exception".
Structures simples	Toute la mise en forme supportée est **simple** sans autre option ni modificateur compliqué. Juste une marque sans aucune option.
facile à apprendre	

	Avec des marques simples et un source lisible, l'apprentissage de txtx2tags est `user friendly`.
De jolis exemples	Les **fichiers d'exemple** inclus dans le package donnent des exemples réels soit simples ou "sur"−compliqués écrits avec txt2tags.
Des outils précieux	Les **fichiers de syntaxe** inclus dans le logiciel (pour les éditeurs vim, emacs, neno et kate) vous aidant à écrire des documents sans erreur de syntaxe.
trois interfaces utilisateur	Il y a **l'interface graphique Tk**, très facile à utiliser, une **interface Web** qui peut être utilisée à distance et une **interface en ligne de commande** pour les connaisseurs des langages de script.
langage script	Avec le mode ligne de commande comportant toutes les options, un utilisateur expérimenté peut **automatiser** les tâches et faire de la **post−édition** sur les fichiers convertis.
Chargez et lancez / Multi−platforme	Txt2tags est un simple **script Python**. Il n'y a pas besoin de compilateur ni de charger des modules supplémentaires. Ainsi il tourne sans problème sur *NIX, Linux, Windows et Macintosh.
Mises à jour fréquentes	Le programme a une mailing list très active avec des utilisateurs qui proposent des corrections et des améliorations. L'auteur lui−même est un utilisateur intensif à la maison et au travail, donc le développement n'est pas prêt de s'arrêter.

Ai−je à payer?

Absolument PAS !

C'est un logiciel libre, GPL, open source et du domaine public. <*mettez−votre−mot−favori−ici*>.

Vous pouvez le copier, l'utiliser, le modifier, le vendre et faire des mises à jour comme s'il vous appartenait. La politique de copyright du logiciel n'est pas un des soucis principaux de l'auteur.

Structures de mise au format supportées

Voici la liste de toutes les structures supportées par txt2tags.

- entête (titre du document, nom de l'auteur, date)
- titre de section (numéroté ou non)
- paragraphes
- modificateurs de fontes
    - ♦ gras
    - ♦ italique
    - ♦ souligné
- fonte non proportionnelle (verbatim)
    - ♦ dans un paragraphe en fonte non proportionnelle
    - ♦ ligne en fonte non proportionnelle
    - ♦ zone en fonte non proportionnelle
- citations
- liens
    - ♦ URL/liens Internet,
    - ♦ liens e−mail,
    - ♦ liens locaux,
    - ♦ liens nommés.
- listes
    - ♦ liste pointée

    - liste numérotée
    - liste de définition
- lignes de séparation horizontales
- image avec un joli alignement
- table (avec ou sans bordures, alignement dans la cellule)
- marques spéciales pour du texte brut (sans traitement)
- macro spéciale pour la date courante (avec une mise en forme souple)
- commentaires (pour des notes, TODO, FIXME)

Cibles supportées

HTML

Tout le monde sait ce qu'est l'HTML. (Internet)

Txt2tags génère des documents HTML propres, qui sont jolis et dont le source est lisible. Il N'UTILISE PAS javascript, des fenêtres ou autre technique futile, qui ne sont pas nécessaires pour des documents simples ou techniques. Mais un fichier CSS peut être utilisé si demandé. Txt2tags génère du code "*HTML 4.0 Transitional*".

Depuis la version 2.0, le code HTML généré est approuvé 100% par le w3c validator.

XHTML

C'est la nouvelle génération de HTML, avec des règles plus strictes aussi près des marques que vous ouvrez. Cela rend le code plus facile à traiter et à comprendre. Pour les cas généraux, considérez que c'est du HTML. Txt2tags génère du code "*XHTML 1.0 Transitional*".

Depuis la version 2.0, le code XHTML généré est approuvé 100% par le w3c validator.

SGML

C'est un format courant de document qui a de puissants outils de conversion. A partir d'un fichier sgml vous pouvez générer des documents html, pdf, ps, info, latex, lyx, rtf. Les outils sgml2* font aussi une table des matières automatique et coupent les sections en sous−pages (sgml2html).

Txt2tags génère des fichiers SGML dans le système de fichiers linuxdoc prêt à être convertis avec les outils sgml2* sans fichier catalogue supplémentaire et sans besoin ennuyeux du SGML

LATEX

Le format académique préféré, bien plus puissant que vous pouvez le rêver. Des livres complets, des formules compliquées et tout texte complexe peuvent être écrits en LaTeX. Mais si vous voulez écrire les marques à la main, préparez vous à vous arracher les cheveux ...

Txt2tags génère des fichiers LaTeX prêts à l'usage, faisant tout le travail complexe de mise en forme avec ses exceptions. L'utilisateur n'a à s'occuper que du texte.

LOUT

Très semblable à LaTeX en puissance, mais avec une syntaxe plus facile, utilisant des "@" à la place des "\" et évitant l'utilisation des accolades dans les cas classiques. Son approche tout−est−objet rend le marquage plus sain.

Txt2tags génère des fichiers Lout prêts à l'usage, qui peuvent être convertis en PS ou PDF en utilisant la commande "lout".

MAN

Les page de manuel UNIX résistent au temps. Les formats de documents passent et elles sont toujours là, inamovibless.

Il y a d'autres outils pour générer des pages de man, mais txt2tags a un avantage : une source de multiples cibles. Ainsi la même page de man peut être convertie en page HTML,

presentation Magic Point, etc.

MGP

Magic Point est un outil de présentation très pratique (genre : Microsoft PowerPoint), qui utilise un langage de marques pour définir tous les écrans. Vous pouvez faire des présentations complexes avec vi/emacs/notepad.

Txt2tags génère des fichiers .mgp prêts à l'usage, définissant toutes les entêtes pour les fontes et pour les définitions de la présentation, iainsi que le support des accents en encodage ISO−8859 avec le support des accents.

POINT FORT 1 : txt2tags crée les fichiers .mgp en utilisant les fontes XFree86 Type1. Ainsi vous n'avez pas à joindre les fichiers de fontes TrueType à votre présentation.

POINT FORT 2 : les définitions de couleurs des fontes sont propres, même sur un système avec une palette de couleurs réduite (même lancé avec la commande `startx -- -bpp 8`) la présentation sera correcte.

Les mots clés sont : convertir et utiliser. Aucun autre besoin ou arrangement nécessaire.

MOIN

Vous ne savez pas ce qu'est MoinMoin ? C'est WikiWiki!

La syntaxe du Moin est un tantinet ennuyeuse quand vous devez mettre `{{{'''''des apostrophes et des accolades'''''}}}`, txt2tags vient avec ses marques simples et sa solution unifiée : un source des cibles multiples.

PM6

J'espère que vous ne le savez pas, mais Adobe PageMaker 6.0 a son propre langage de balises. Les styles, les couleurs de table, les modificateurs de caractères et la majorité des possibilités à la souris sont également disponibles dans son langage de marques. Vous avez juste besoin d'accéder au menu "Import tagged text". Juste pour information, c'est un langage de marques <comme HTML>.

Txt2tags génère toutes les marques et définit déjà une entête complète et fonctionnelle, réglant la mise au format et les styles de paragraphe. C'est le morceau le plus difficile.

ATTENTION: Pas de coupure de ligne. Un paragraphe doit être sur une seule ligne.

Note de l'auteur : *Mon livre entier en portugais regular expression's book* //a été écrit avec VI, convertit en PageMaker avec txt2tags et envoyé *sous presse.*

TXT

TXT est le texte. Le seul vrai type de mise au format. Quoique les marques de txt2tags soient intuitives et discrètes, vous pouvez les enlever en convertissant le fichier en texte pur.

Les titres sont soulignés. Et le texte est laissé comme dans le fichier source.

Status des structures supportées par cible

Structure	html	xhtml	sgml	tex	lout	man	mgp	moin	pm6	txt
entêtes	O	O	O	O	O	O	O	N	N	O
entête de section	O	O	O	O	O	O	O	O	O	O
paragraphes	O	O	O	O	O	O	O	O	O	O
gras	O	O	O	O	O	O	O	O	O	−
italique	O	O	O	O	O	O	O	O	O	−
souligné	O	O	−	O	O	−	O	O	O	−

fonte non proportionnelle	O	O	O	O	O	−	O	O	O	−
ligne verbatim	O	O	O	O	O	O	O	O	O	−
zone verbatim	O	O	O	O	O	O	O	O	O	−
zone de citation	O	O	O	O	O	O	O	O	O	O
liens internet	O	O	O	−	−	−	−	O	−	−
liens e−mail	O	O	O	−	−	−	−	O	−	−
lens locaux	O	O	O	N	N	−	−	O	−	−
liens nommés	O	O	O	−	−	−	−	O	−	−
liste pointée	O	O	O	O	O	O	O	O	O	O
liste numérotée	O	O	O	O	O	O	O	O	O	O
liste de definition	O	O	O	O	O	O	N	N	N	O
ligne horizontale	O	O	−	O	O	−	O	O	N	O
image	O	O	O	O	O	−	O	O	N	−
table	O	O	O	O	N	O	N	O	N	N
Extras	**html**	**xhtml**	**sgml**	**tex**	**lout**	**man**	**mgp**	**moin**	**pm6**	**txt**
alignement d'image	O	O	N	N	O	−	O	N	N	−
alignement cellule table	O	O	O	O	N	O	N	O	N	N

Légende

O *supporté*

N *non supporté (peut−être dans une révision future)*

− *non supporté (ne peut pas l'être dans cette cible)*

Les trois interfaces utilisateur : GUI, internet et ligne de commande

Comme les utilisateurs ont des besoins différents et des environnements divers, txt2tags est très souple dans son utilisation.

Il y a trois interfaces utilisateur pour le programme, chacune avec ses particularités et fonctionnalités.

- **GUI**: écrite en Tk, apporte le fenêtrage et le clic de souris à txt2tags
- **Web**: écrite en PHP, permet de lancer txt2tags sur un butineur, ne nécessitant aucune installation côté client.
- **ligne de commande**: écrite en Python, c'est le coeur du programme. Toutes les options sont disponibles en option.

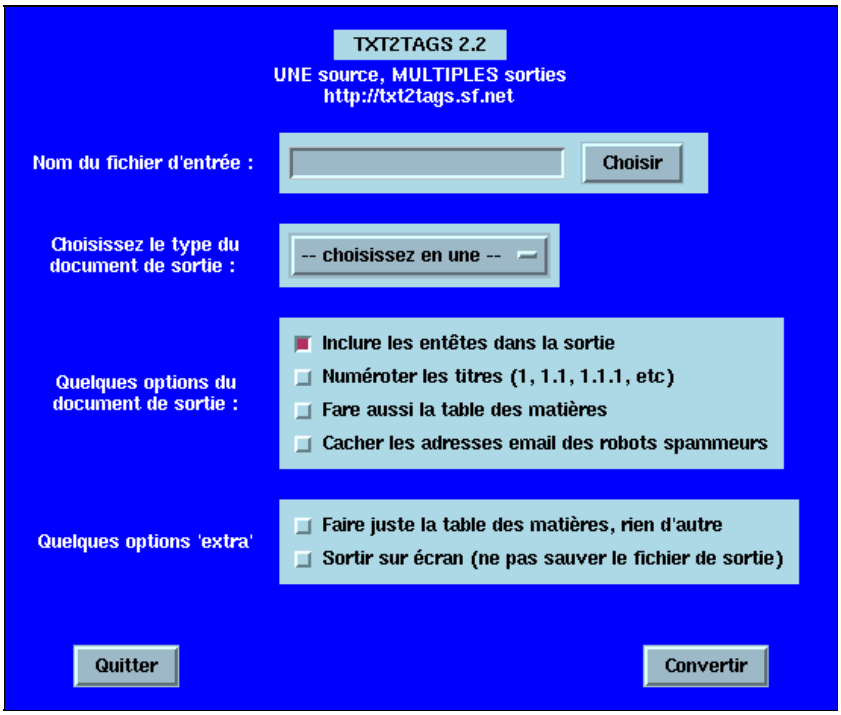
Interface graphique Tk

Depuis la version 1.0, il y a une jolie interface graphique, qui fonctionne sur Linux, Windows, Mac et autres.

Le programme détecte automatiquement si votre système peut afficher l'interface, et est lancé sans argument. On peut forcer l'interface graphique avec l'option `−−gui`. S'il manque une ressource, le programme le dira.

Note: Le module Tkinter est nécessaire. Comme il est disponible dans la distribution standard de Python, vous devez certainement l'avoir.

L'interface est simple et intuitive.



1. Vous localisez le fichier .t2t sur le disque et les options sont chargées.
2. Si aucune cible n'est définie, vou devez en choisir une
3. Ensuite vous pouvez choisir quelques options, mais aucune n'est nécessaire
4. A la fin vous appuyez sur le bouton "convertir"

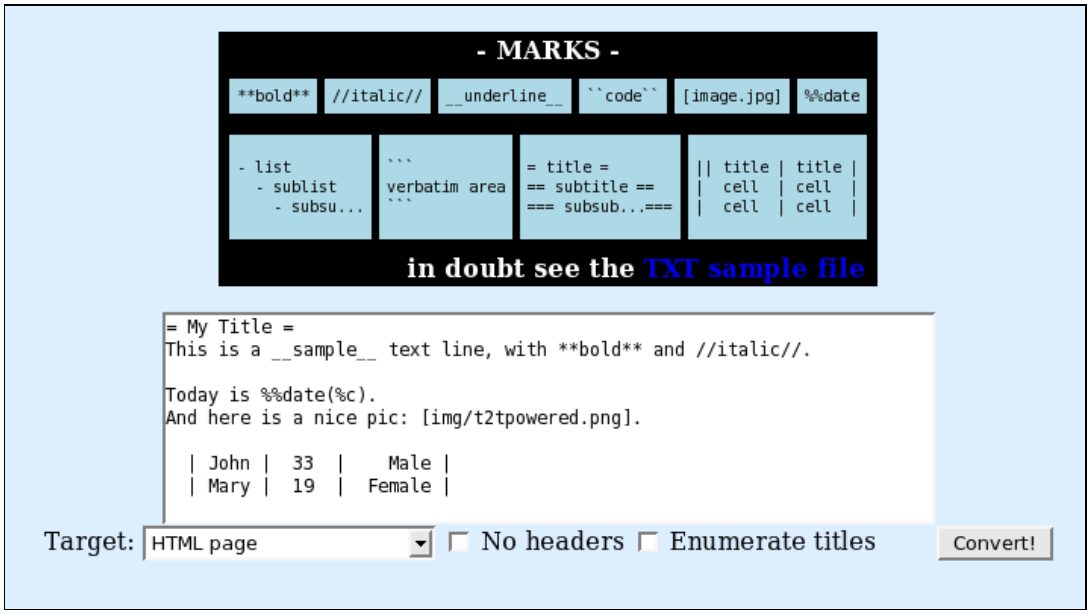
Un option pratique est "*visualisation à l'écran*". Vous pouvez voir le code généré dans une autre fenêtre , aucun fichier n'est sauvegardé. Quand le code est correct, vous enlevez l'option et la conversion s'effectuera normalement.

Les couleurs par défaut peuvent être changées dans le fichier `~/.txt2tagsrc`, en utilisant les réglages `%!guicolors`. Par exemple :

```
% choisir mes couleurs pour l'interface graphique (bg1, fg1, bg2, fg2)
%!guicolors: blue white brown yellow
```

Interface Web

L'interface Web est disponible à l'adresse http://txt2tags.sf.net/online.php. Vous pouvez tester et utiliser le programme avant de le charger.



On peut aussi installer cette interface sur le réseau local. Cela évite d'avoir à l'installer sur toutes les machines.

Interface ligne de commmande

Pour les utlisateurs en ligne de commande l'option −−help doit suffire.

```
Utilisation : txt2tags [OPTIONS] [entrée.t2t ...]

  -t, --target              indiquez le type de document de sortie supporté :
                        html, xhtml, sgml, tex, lout, man, mgp, moin, pm6, txt
  -i, --infile=ENTREE   choisissez le fichier d'entrée ('-' pour STDIN)
  -o, --outfile=SORTIE  choisissez le fichier de sortie ('-' pour STDOUT)
  -n, --enum-title      numéroter les titres en : 1, 1.1, 1.1.1, etc
  -H, --no-headers      supprimer les entêtes, les titres et les pieds de page
      --headers         montrer les entêtes, les titres et les pieds de page (par défaut)
      --encoding=ENC    choisir l'encodage de sortie (utf-8, iso8859-1, etc)
      --style=FILE      utiliser FILE comme feuille de style (comme HTML CSS)
      --css-sugar       insère des étiquettes CSS pour des  sorties HTML et XHTML
      --css-sugar       insèrer le contenu du fichier CSS pour des sorties HTML et XHTML
      --mask-email      cacher l'email des robots spammeurs x@y.z devient x (a) y z
      --toc             ajouter la table des matières à la sortie
      --toc-only        imprime la table des matières et sort
      --toc-level=N     limiter la profondeur de la table des matières à N
      --rc              lire le fichier de configuration ~/.txt2tagsrc (option par défaut)
      --gui             appeler l'interface graphique Tk
  -q, --quiet           mode silencieux, plus de sortie (sauf les erreurs)
  -v, --verbose         imprimer des messages d'information pendant la conversion
  -h, --help            imprimer cette aide et sortir
  -V, --version         imprimer la version et sortir
      --dump-config     imprimer toute la configuration trouvée et sortir

Désactive les options (OFF)
     --no-outfile, --no-infile, --no-style, --no-encoding, --no-headers
     --no-toc, --no-toc-only, --no-mask-email, --no-enum-title, --no-rc
     --no-css-sugar, --no-css-inside, --no-quiet

Exemple :
    txt2tags -t html --toc myfile.t2t

Par défaut, la sortie convertie est sauvée dans 'infile.<type>'
Utilisez --outfile pour imposer un fichier de sortie
Si le fichier d'entrée est '-', la lecture est à partir de STDIN
```

```
Si le fichier de sortie est '−', la sortie est sur STDOUT
```

Exemples

Si vous avez créé un fichier marqué `file.t2t`, voici quelques exemples :

Convertir en HTML	`$ txt2tags −t html file.t2t`
Le même avec redirection	`$ txt2tags −t html −o − file.t2t > file.html` .
Avec une table des matières	`$ txt2tags −t html −−toc file.t2t`
Et aussi avec les titres numérotés	`$ txt2tags −t html −−toc −−enum−title file.t2t`
vue rapide de la table des matières	`$ txt2tags −−toc−only file.t2t`
la même numérotée	`$ txt2tags −−toc−only −−enum−title file.t2t` .
coder une ligne à partir de STDIN	`$ echo −e "\n**bold**" | txt2tags −t html −−no−headers −`
Tester l'option masquer les emails	`$ echo −e "\njohn.wayne@farwest.com" | txt2tags −t txt −−mask−email −−no−headers −`
Edition post−conversion	`$ txt2tags −t html −o− file.t2t | sed "s/<BODY .*/<BODY BGCOLOR=green>/" > file.html`

Note

Depuis la version 1.6 vous pouvez faire du traitement avant et après conversion avec les filtres `%!preproc` et `%!postproc`.

Deuxième partie − OK je le veux, que faire maintenant ?

Téléchargez le programme et lancez le sur votre machine.

Charger et installer Python

Avant tout, vous devez charger et installer l'interpréteur Python sur votre système. Si vous l'avez déjà, sautez ce paragraphe.

Python est un bon programme, il fonctionne sur Windows, Linux, UNIX, Macintosh et autres et peut être téléchargé à partir du site web Python. Les instructions d'installation sont disponibles sur le site web. Txt2tags marche mieux avec une version de Python 1.5 ou supérieure.

Si vous n'êtes pas sûr d'avoir Python, ouvrez une console (tty, xterm ou MSDOS) et tapez `python`. Si ce n'est pas installé, votre système vous le dira.

Charger txt2tags

Le site officiel de la distribution txt2tags est http://txt2tags.sf.net/src.

Tous les fichiers sont des tarballi (fichiers .tgz), qui peuvent être décompressés par la plupart des utilitaires (Winzip inclus).

Prenez la **dernière** version (la plus récente ou le numéro de version le plus élevé). Les versions précédentes restent pour des raisons historiques.

Installer txt2tags

Comme tous les scripts Python, txt2tags ne nécessite aucune installation.

Le seul fichier nécessaire pour le programme est le script txt2tags. Les autres fichiers du tarball sont de la documentation, des outils et des fichiers d'exemple.

Le moyen le plus sur de lancer txt2tags est de l'appeler par Python.

```
prompt$ python txt2tags
```

Si vous voulez installer txt2tags sur le système comme programme, copiez (ou faites un lien) sur le fichier script txt2tags dans un répertoire référencé par la variable PATH et assurez vous que le système peut le lancer.

UNIX/Linux
: Rendez le script exécutable (`chmod +x txt2tags`) et copiez le dans un répertoire référencé par la variable PATH (`cp txt2tags /usr/local/bin`).

Windows
: Renommez le fichier script en ajoutant l'extension .py (`ren txt2tags txt2tags.py`) et copiez le dans un répertoire de la variable PATH (`copy txt2tags.py C:\WINNT`).

Après cela, vous pouvez créer un icone pour votre bureau, si vous voulez utiliser l'interface graphique.

Paquets spéciaux pour les utilisateurs Windows

Il y a aussi deux distributions .EXE pour txt2tags, chacune installe le programme sur les machines Window avec quelques clics :

- Le script txt2tags pour ceux qui ont l'interpréteur Python déjà installé
- La version complète, qui n'a pas besoin de l'interpréteur Python (il y a une version légère de Python incluse).

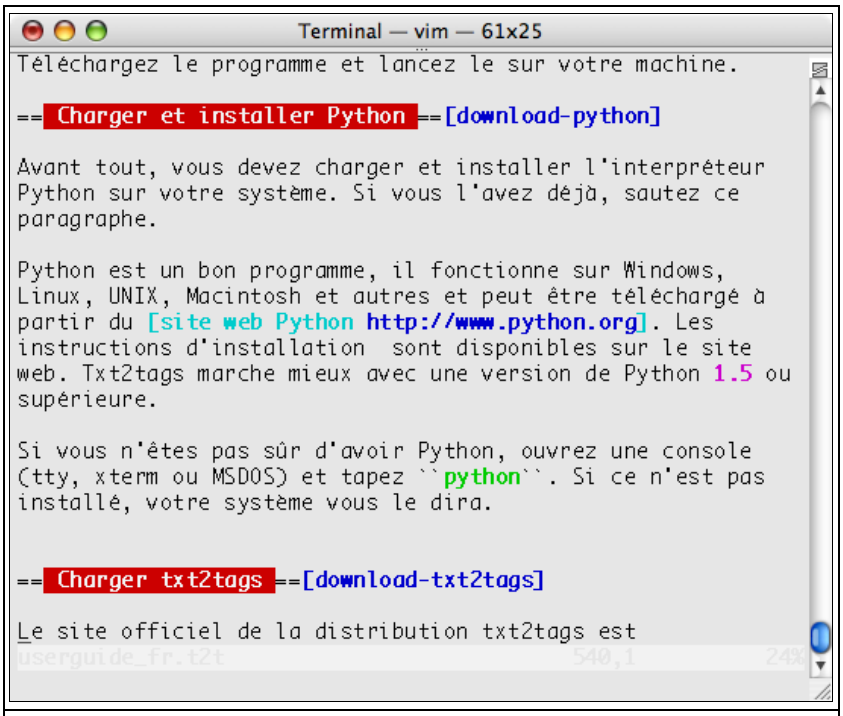
Visitez le site *Txt2tags−Win* pour charger les paquets : http://txt2tags−win.sf.net

Installer les colorateurs syntaxiques pour les éditeurs

Txt2tags est livré avec des fichiers de colorateurs syntaxiques utilisables avec les éditeurs suivants :

- Vim (www.vim.org)
- Emacs (www.emacs.org)
- Nano (www.nano−editor.org)
- Kate (http://kate.kde.org)

Ces fichiers de coloration syntaxique connaissent toutes les règles et les marques de txt2tags, aidant l'utlisateur à écrire des documents sans faute de syntaxe. Visualisant les marques en couleur, vous voyez en direct si vous écrivez correctement.



Exemple de fichier ouvert avec l'éditeur Vim

Chaque éditeur a une procédure d'installation spécifique, lire l'entête du ficher de syntaxe et la documentation de l'éditeur.

Troisième partie − Ecrire et convertir son premier document

Tester les outils

Pour faire votre première conversion, vous avez besoin de trois choses : txt2tags, un éditeur de texte et un butineur.

1. Vérifiez que txt2tags fonctionne sur votre machine.
    - **Interface en ligne de commande :** Appelez "txt2tags" sur une ligne de commande et le programme va vous donner le message "Missing input file" ou similaire. Si cela ne marche pas essayez "python /chemin/vers/txt2tags" ou "/chemin/vers/python /chemin/vers/txt2tags" si Python n'est pas dans votre PATH.
    - **Interface Gui :** Cliquez sur l'icone du programme pour lancer l'interface graphique.
2. Ouvrez un éditeur de texte que vous connaissez bien. cela peut être **n'importe lequel**, de vi jusqu'à MS Word ou Open Office. Créez un nouveau document marqué qui sera votre premier document txt2tags.
3. Lancez votre butineur préféré pour voir les résultats de la conversion dans une page HTML.

Ecrire l'entête du document

1. Allez dans l'éditeur de texte et tapez sur la **première** ligne du document *Mon premier doument*
2. Sur la deuxième ligne faire un sous titre avec le texte *test txt2tags*
3. Puis sur la troisième ligne mettre une information de date *Lundi 2029*

Si tout c'est bien passé vous avez un document avec le contenu suivant :

```
Mon premier document
test txt2tags
Lundi 32 Janvier 2029
```

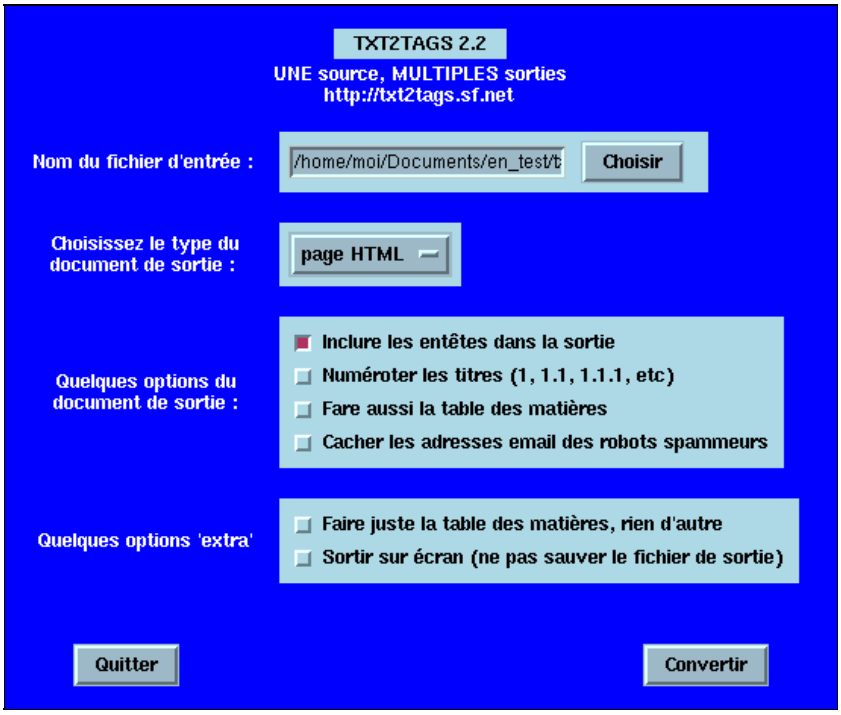
Ceci n'est qu'une partie du document, mais vous pouvez le convertir et tester le résultat.

Maintenant sauvegardez le résultat avec le nom `test.txt`. Notez dans quel dossier vous sauvez le fichier, vous en aurez besoin plus tard.

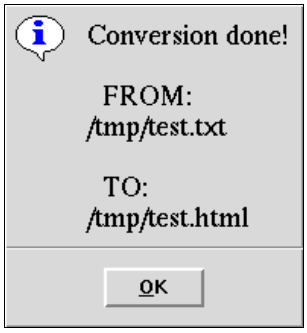
La première conversion − l'interface graphique

Si vous utilisez l'interface en ligne de commande passez au paragraphe suivant.

Si vous utilisez l'interface graphique procédez comme suit:



1. Appuyez sur le bouton "choisir" et choisir le fichier `test.txt` que vous venez de sauver (souvenez vous du dossier !)
2. De retour à la première image, choisir "page HTML" pour "Choisissez le type du document de sortie"
3. Appuyez sur le bouton "convertir"



Une boite de dialogue apparaît, qui vous informe que le fichier a été converti avec succès. Notez que le fichier HTML généré a été sauvé dans le même dossier que le fichier texte avec l'extension "html".

La première conversion − interface ligne de commande

Si vous utilisez l'interface graphique allez au paragraphe précédent.

Si vous êtes dans l'interface ligne de commande allez dans le dossier où a été sauvé votre fichier et tapez la commande :

```
txt2tags  --target  html  test.txt
```

Notez qu'il y a des espaces entres les parties de la commande mais pas dans la chaine "−−target", c'est une option. Cette option est suivie de la chaine "html", qui dit au programme dans quel format votre fichier texte doit être converti. Le dernier argument est le nom du fichier.

Si la conversion se réalise, le résultat est sauvé dans le fichier `test.html` et le programme affichera

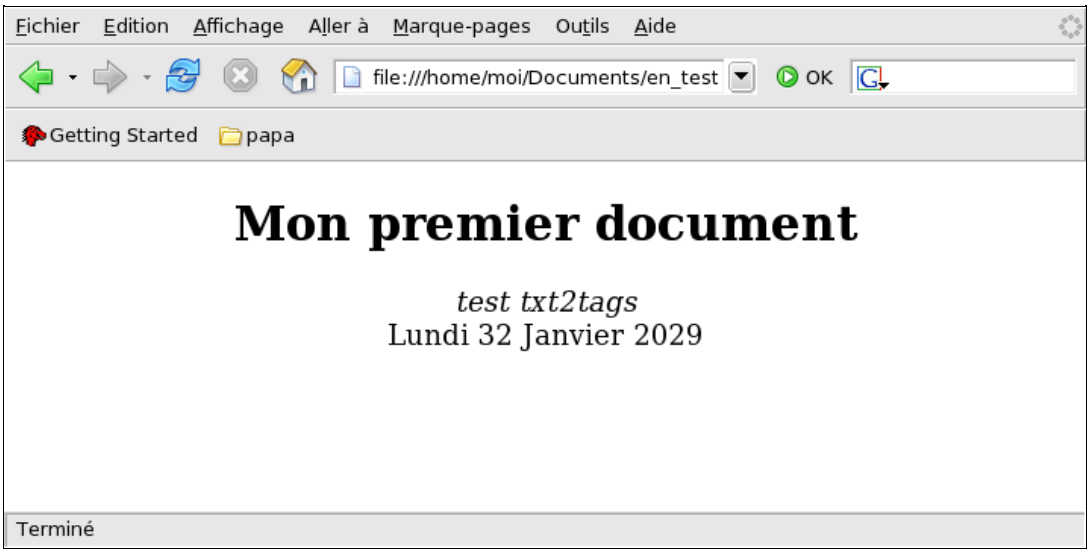
le message "*txt2tags écrit test.html*" Sinon il vous indiquera l'erreur que vous avez faite en tapant la ligne de commande. Vérifiez attentivement.

Voici un exemple de ce que vous verrez à l'écran :

```
prompt$ txt2tags --target html test.txt
txt2tags écrit test.html
promtp$
```

Tester le résultat

Ouvrez le fichier `test.html` avec le butineur pour vérifier que tout est bon.



On y arrive ! Vous avez juste tapé trois lignes simples de texte et txt2tags a fait tout le travail pour configurer l'entête de la page HTML. L'alignement du texte, les tailles, les espacements et l'apparence. Voyez que le titre principal est également dans l'onglet du butineur.

Vous écrivez du texte, txt2tags fait le reste :)

Astuce : Vous pouvez également utiliser des CSS sur les pages HTML générées par txt2tags, ainsi l'apparence de la page est configurable à 100%.

Ecrire le corps du document

Retournons à l'éditeur de texte, l'étape suivante est d'écrire le corps du document. Vous pouvez écrire du texte brut comme vous le faites pour les emails. Vous voyez que txt2tags reconnait les paragraphes et les listes d'items automatiquement, vous n'avez pas besoin de les "marquer".

Sauvez à nouveau, convertissez et testez le résultat. C'est le cycle de développement avec txt2tags. Vous ne vous intéressez qu'au contenu du document, terminant les documents plus rapidement qu'avec les autres éditeurs. Pas de cliquage de souris, pas de menus, de fenêtre ni de distraction.

Regardez le contenu du fichier suivant `test.txt`, qui n'est que du texte brut et comparez le avec la page HTML générée :

```
Mon premier document
test txt2tags
Lundi 32 Janvier 2029

Essayons donc ce truc qui s'appelle txt2tags.
```

```
Ce que je met c'est juste pour noircir l'écran.
C'est pas un temps à aller dehors chercher :
- des mouches
- des souris
- des canards
- des éléphants
- des baleines
```

Vous pouvez écrire une page complète sans aucune connaissance de HTML. Vous n'avez pas besoin d'insérer des marques. En plus le même fichier texte peut être converti dans tous les autres formats cibles supportés par txt2tags.

A côté du texte brut, txt2tags a quelques marques très simples, que vous utilisez quand vous avez besoin d'autres mises au format ou structures comme le gras, l'italique, les titres, les images, les tables et autres. Un exemple rapide : `**commencez en gras**` et `== les signes == pour le titre ==`. Vous pouvez apprendre les marques grace au fichier Txt2tags Markup Demo.

Quatrième partie − maitriser les concepts de txt2tags

Les zones d'un document .t2t

Les fichiers marques txt2tags sont divisés en 3 zones. Chacune a ses propres règles et usage. Ce sont :

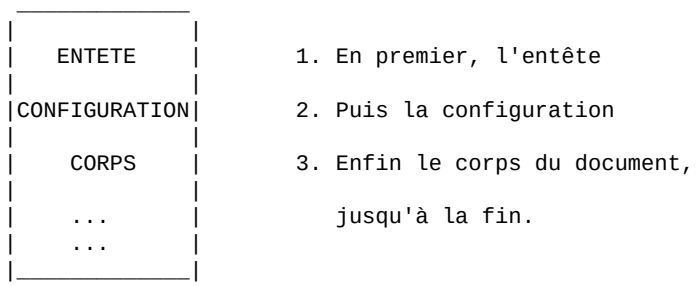
ENTETE
: La zone pour le titre du document, l'auteur, la version et la date (ceci est optionnel)

CONFIGURATION
: La zone pour la configuration générale du document et le comportement de la conversion(ceci est optionnel).

CORPS
: La place du contenu du document (obligatoire)

Comme vu au dessus, les deux premières zones sont optionnelles, seul le *CORPS* est obligatoire.

Les zones sont délimitées par des règles spéciales, qui seront vues en détail dans le chapitre suivant. La représentation graphique des zones du document est la suivante :



En bref, les zones sont définies de la façon suivante :

Entête	Les 3 premières lignes du fichier ou la première ligne vide
Configuration	Commence après l'entête (4e ou 2e ligne) et termine lorsque le corps commence
Corps	La première ligne de texte valide (pas un commentaire ou une configuration après l'*ENTETE*

Exemple complet

```
My nice doc Title
Mr. John Doe
Last Updated: %%mtime(%c)

%! Target  : html
%! Style   : fancy.css
%! Encoding: iso-8859-1
%! Options : --toc --enum-title

Hi! This is my test document.
Its content will end here.
```

ENTETE

Localisation:

- position fixe : **les 3 premières lignes** du fichier. C'est tout.
- position fixe : **la première ligne** du fichier si elle est blanche. Cela signifie une entête vide.

L'ENTETE est la seule zone qui a une position fixe, et est orientée ligne. Elle est localisée dans les trois premières lignes du fichier source.

Ces lignes sont libres, aucune information statique de type est demandée, mais le contenu suivant est recommandé :

- *ligne 1*: titre du document
- *ligne 2*: nom de l'auteur et/ou email
- *ligne 3*: date du document et/ou version (un lieu privilégié pour `%%date`)

Gardez en mémoire que les trois premières lignes du document source seront les trois premières lignes du ducument cible, séparées et contrastées par rapport au corps du document (grandes lettres en gras). Si la pagination est autorisée, l'entête sera seule et centrée sur la première page.

Plus (ou pas) de lignes d'entête

Quelquefois l'utilisateur désire spécifier moins de trois lignes pour l'entête, ne donnant que le titre et/ou la date.

Il n'y a qu'à laisser la deuxième et/ou la troisième ligne vide et cela ne sera pas inclus dans le document final. Mais gardez en mémoire que même vides, ces lignes font partie de l'entête et que le corps du document **doit** commencer après cette troisième ligne.

Le titre est le seul élément demandé de l'entête, mais si vous le laissez vide, vous dites que votre document n'aura **pas d'entête**. le corps du document commencera juste après à la deuxième ligne.

Ne pas mettre d'entête est souvent utile si vous voulez spécifier votre propre entête personnalisée après conversion. L'option de ligne de commande `−−no−headers` est nécessaire pour cette opération.

En deux mots

En résumé : "les entêtes sont des positions pas des contenus."

Placez un texte sur la première ligne et il apparaitra sur la première ligne du fichier résultat. Idem pour les deuxième et troisième lignes de l'entête.

CONFIGURATION

Localisation:

- Commence juste après l'ENTETE
  - ♦ Commence à la **quatrième ligne** du fichier si des **entêtes** sont spécifiées.
  - ♦ Commence à la **deuxième ligne** du fichier si **aucune entête** n'est spécifiée.
- Se termine lorsque le CORPS commence.
  - ♦ Se termine par une ligne vide, une ligne qui n'est pas un réglage, ou une ligne de commentaire

La zone de CONFIGURATION est optionnelle. Un utilisateur moyen peut écrire un tas de fichiers txt2tags sans savoir que cela existe, mais les utilisateurs expérimentés apprécieront la puissance et le contrôle que cela procure.

La zone de CONFIGURATION est utilisée pour ranger les réglages du document, vous n'aurez pas à taper les options dans la ligne de commande au moment de la conversion. Par exemple vous pouvez définir le format de la cible et l'encodage.

Lire la section réglages pour plus d'informations.

CORPS

Localisation:

- Commence à la première ligne valide de texte du fichier.
    - ♦ Les entêtes, les réglages et les commentaires ne sont **pas** des lignes de texte valides.
- Se termine à la fin du fichier (EOF).

Le CORPS est tout ce qui n'est pas ENTETE ou CONFIGURATION.

Le CORPS est constitué du contenu du document et de toutes les structures et mises en forme que txt2tags peut reconnaitre. Dans le corps vous pouvez inclure des commentaires pour les *TODO* et les *notes personnelles*.

Vous pouvez utiliser l'option en ligne de commande `--no-headers` pour convertir le corps du document, en supprimant les entêtes. C'est pratique pour fabriquer vos entêtes dans un fichier séparé, que vous concaténez ensuite.

Réglages

Les réglages sont des lignes de commande placées dans la CONFIGURATION et qui modifient la conversion. Leur syntaxe est :

%! clé : valeur

Liste des clés valides :

clé	Description
Target	définit la cible par défaut dans laquelle le document sera converti
Options	Définit les options par défaut qui seront utilisées à la conversion. Le format est le même que dans la ligne de commande.
Style	Définit le style du document. Utilisé pour définir un fichier CSS pour HTML/XHTML et pour charger un package LaTeX.
Encoding	Choisit le jeu de caractères. Utilisé si le document contient des caractères accentués ou des caractères non ASCII
PreProc	Filtre d'entrée. Définit les règles "trouve et remplace" qui seront appliquées au document source.
PostProc	filtre de sortie. Définit les règles "trouve et remplace" qui seront appliquées au document de sortie.

Exemple:

```
%! Target  : html
%! Options : −−toc −−toc−level 3
%! Style   : fancy.css
%! Encoding: iso−8859−1
%! PreProc : "AMJ"        "Aurelio Marinho Jargas"
%! PostProc: '<BODY.*?>'  '<BODY bgcolor="yellow">'
```

Option en ligne de commande

Le moyen le plus rapide de changer le comportement par défaut est d'utiliser les options de la ligne de commande. Cela n'est disponible que sur l'interface en ligne de commande pas sur l'inferface GUI ou Web.

Comme les autres programmes système, le programme accepte un jeu d'options prédéfinies. Une option est un tiret suivi par une lettre ou deux tirets suivis d'un ou plusieurs mots comme `−t` ou `−−target`. A propos de l'option "target", c'est la seule obligatoire, toutes les autres sont optionnelles.

Les options généralement utilisées sont `−−outfile` pour définir un fichier de sortie particulier, `−−toc` pour activer la génération automatique de la table des matières et `−−encoding` pour choisir le jeu de caractères. La plupart des options peuvent être désactivées en la précédant de "no−" devant, par exemple :`−no−encoding` et `−−no−toc`.

Vous pouvez enregistrer les options désirées dans le fichier source dans la CONFIGURATION, en utilisant le réglage `%!option`. Par ce moyen, vous n'avez plus à les retaper dans la ligne de commande. Exemple: `%!options: −−toc −o mydoc.html`. L'exception est la spécification de la cible qui a sa propre syntaxe : `%!target: html`.

Utilisez l'option `−−help` pour avoir la liste complète des options disponibles dans txt2tags.

Fichier de configuration utlisateur (fichier RC)

Le fichier de configuration utilisateur (appelé aussi fichier RC) est le lieu de stockage des réglages qui vont être gérés pour **tous** les fichiers convertis. Si vous insérez les mêmes réglages pour tous les fichiers .t2t que vous écrivez, déplacez les dans le fichier RC, ils seront utilisés globalement pour les fichiers existants et à venir.

L'emplacement par défaut de ce fichier dépend de votre système. Il peut aussi être spécifié par l'utilisateur, en utilisant une variable d'environnement.

	localisation du fichier RC
Windows :	`%HOMEPATH%\_t2trc`
Linux et autres :	`$HOME/.txt2tagsrc`
défini par l'utilisateur :	variable d'environnement `T2TCONFIG`

Le format des réglages est exactement le même que celui utilisé dans la zone CONFIGURATION des fichiers .t2t. Il y a un exemple de fichier RC dans le tarball à `doc/txt2tagsrc`. Exemple:

```
% ma config

%%% utiliser un fichier CSS pour le HTML
%!options(html): −−css−suggar

%%% changer la profondeur de la table des matières à 4 pour toutes les cibles
%!options: −−toc−level 4

%%% encodage par défaut pour tous les documents
```

```
%!options: −−encoding iso−8859−1
```

Toute ligne non vide, un commentaire ou une ligne de configuration valide va générer une erreur à l'exécution de txt2tags. Donc soyez attentif quand vous éditez ce fichier.

Txt2tags applique automatiquement le fichier RC à tout fichier qu'il convertit. Si vous voulez désactiver ce comportement pour un fichier spécifique, utilisez l'option `−−no−rc` en option dans la ligne de commande.

Ordre de chargement des configuration et précédence

Il y a trois moyens pour préciser à txt2tags les options et les réglages à utiliser. Ceci est l'ordre dans lequels ils sont lus et appliqués :

1. Le fichier de configuration utilisateur (RC)
2. La zone CONFIGURATION dans le document source
3. Les options de la ligne de commande

En premier txt2tags lit le contenu du fichier RC s'il existe et applique contenu sur le fichier source. Puis il scrute la zone CONFIGURATION du fichier source courant si elle existe et l'applique en surchargeant les règles du fichier RC en cas de conflit. En dernier, il applique les options de la ligne de commande qui ont toujours priorité.

Ainsi, si l'encodage est défini dans les trois sources, ce sera l'option de la ligne de commande qui sera utilisée.

commande %!include

La commande `include` est utilisée pour inclure un fichier externe dans le corps du document source courant. Ce n'est pas une configuration mais une commande qui est valide dans le CORPS du document.

La commande `include` est pratique pour découper un gros document en d'autres plus petits (comme les chapitres d'un livre) ou pour inclure le contenu complet d'un fichier externe dans le document source. Exemple :

```
Mon premier livre
Dr. John Doe
1ere Edition

%!include: intro.t2t
%!include: chapter1.t2t
%!include: chapter2.t2t
...
%!include: chapter9.t2t
%!include: ending.t2t
```

Indiquez juste le nom du fichier après la chaine `%!include`. Les spécification optionnelles de la cible sont supportées. La commande suivante est valide :

```
%!include(html): file.t2t
```

Notez que la commande include insert le CORPS dans le fichier source du document. Les zones ENTETE et CONFIGURATION sont ignorées. Cela permet de convertir le fichier inclus séparément ou avec le document principal.

Il y a 3 variantes de la commande include :

- inclusion Verbatim
- inclusion tel−quel (raw)
- inclusion marquée

Le type **verbatim** inclut le fichier texte en préservant le format et les espaces du fichier original comme s'il était marqué par VERB(```). Pour cela encadrer le nom de fichier avec des apostrophes inverses :

```
%!include: ``/etc/fstab``
```

Le type **tel−quel** inclut un fichier texte tel quel, ne cherchant pas les marques txt2tags et **ne les interprétant pas**, comme s'il était marqué par RAW ("""). Pour cela encadrer le nom de fichier par des apostrophes doubles comme :

```
%!include: ""nice_text.txt""
```

Le type **marqué** est envoyé directement dans le fichier résultat, sans aucun traitement effectué par txt2tags. Par ce moyen vous pouvez inclure des parties marquées ou du code avec des marques de sortie plus complexes non supportées par txt2tags :

```
%!include(html): ''footer.html''
```

Notez que le fichier est encadré par des apostrophes, et comme le texte est inclus déjà marqué vous devez spécifier la cible pour éviter les erreurs.

commande %!includeconf

La commande `includeconf` est utilisée pour inclure la configuration d'un fichier externe dans le fichier courant. Cette commande n'est valide que dans la zone CONFIGURATION.

C'est pratique pour avoir la même configuration pour de multiples fichiers, vous pouvez en centraliser la gestion. Dans le fichier où vous voulez inclure cette confuguratiion globale, mettez un appel `includeconf`. Par exemple :

```
Mon premier Document
John Doe
Juillet 2004

%!includeconf: config.t2t

Hi, ceci est mon premier document
```

Le format dans le fichier inclus est le même que dans le fichier RC.

Cinquième partie − maitriser les marques

Synthèse de toutes les marque stxt2tags :

Basic	**. . . . . .**	**modificateurs**	**. . . . . .**
Entête	3 premières lignes	*gras*	**mots**
Titre	= mots =	*Italique*	//mots//
2cwtitre numéroté	+ mots +	*souligné*	__mots
Paragraphe	mots	*fonte non proportionnelle*	``mots``
blocs de texte	**. . . . . .**	**autre**	**. . . . . .**
Quote	<TAB>mots	*ligne de séparation*	−−−−−−−−−−...
Liste	− mots	*ligne épaisse*	==========...
liste numérotée	+ mots	*liens*	[label url]
liste de définition	: mots	*Image*	[filename.jpg]
ligne Verbatim	``` mots	*Commentaire*	% commentaire
zone Verbatim	```\n lignes \n```	*texte brut*	""mots""
ligne tel−quel(raw)	"""" mots	*Table*	| cell1 | cell2 | cell3...
zone tel−quel (Raw)	""""\n lignes \n""""	*Ancre*	= titre =[ancre]

Règles générales :

- **Entêtes** Ce sont les 3 premières lignes du document, les marques ne sont pas interprétées.
- **Titres** ce sont des caractères "=" ou "+" équilibrés autour du texte du titre. Plus il y a de caractères, plus le niveau du titre est important.
- les **modificateurs** n'acceptent pas d'espace entre les marques et leur contenu.
- les marques de **commentaire** "%" doivent être au début de la ligne (première conlonne).
- les fichiers **Image** doivent se terminer par GIF, JPG, PNG ou similaire.
- Les seules marques **multiligne** sont les marques des zones Verbatim et tel−quel (raw).
- **Aucune marque n'est interprétée** dans les zones Verbatim et tel−quel (raw).
- Les **lignes séparatrices** doivent avoir au moins 20 caractères.
- La profondeur des citations et les listes est définie par **l'indentation**.
- Les **titres de table** sont définis par deux "||" au début de la ligne.

Entête

- **Description :** Identifie l'entête du document
- **Propriétés :** Multiline, FreeSpaces, !Align, !Nesting
- **Contenu :** Macros
- **Syntaxe :**
  - Les 3 premières lignes du fichier source.
  - Pour ne pas spécifier d'entête laisser la première ligne vide Pratique pour les entêtes particulières et les spécialistes "oneliners" en ligne de commande
  - Laisser les deuxième et troisième lignes vides pour éliminer les autres parties de l'entête.
- **Détails :**
  - Les marques ne sont pas interprétées
  - les 3 lignes doivent être les trois premières du document final, elles vont être mises en valeur par rapport au corps du document et isolées sur la première page (si la pagination est autorisée).

- Le contenu des entêtes est quelconque, sans aucune obligation de format, mais le contenu recommandé pour la plupart des documents est :
  - ◊ Ligne 1: Titre du document
  - ◊ Ligne 2: Nom de l'auteur et/ou email
  - ◊ Ligne 3: Date du document et/ou version (le bon endroit pour %%mtime)

Titre, titre numéroté

- **Description :** Identifie un titre de section (numéroté ou non)
- **Propriétés :** !Multiline, FreeSpaces, !Align, !Nesting
- **Contenu :** Raw
- **Syntaxe :** − Pour des titres numérotés changez "=" par "+" sur les règles suivantes
  - des signes équilibrés autour `=comme ceci=`
  - Plus il y a de signes, plus le niveau est élevé : `=titre=`, `==soustitre==`, `===soussoustitre===`, ...
  - Il ya a un maximum de 5 niveaux `===== comme ceci =====`
  - Des signes non équilibrés ne font pas un titre , `=comme ceci===`
  - Des espaces libres dans les marques ne sont pas autorisés `= comme ceci =`
  - Les titres peuvent avoir des ancres `=comme ceci=[ancre]`. Pour créer une ancre créez un `[lien local#ancre]`
- **Détails :**
  - Les marques ne sont pas interprétées
  - Les macros ne sont pas interprétées

Paragraph

- **Description :** Identifie un paragraphe de texte
- **Propriétés :** Multiline, FreeSpaces, !Align, !Nesting
- **Contenu :** Macros, Beautifiers, Raw, Links, Image, Comment
- **Syntaxe :**
  - Les paragraphes sont des groupes de lignes délimités par des lignes vides
  - D'autres blocs comme les listes, les citations, les tables et les zones verbatim terminent aussi un paragraphe.

Commentaire

- **Description :** Utilisé pour indérer du texte qui n'apparaîtra pas dans le document cible
- **Propriétés :** !Multiline, !<FreeSpaces, !Align, !Nesting
- **Contenu :** −
- **Syntaxe :**
  - Un ligne avec le caractère "%" en première colonne `% comme ceci`
  - PAS d'espace devant
- **Détails :**
  - Comme les commentaires, ils ne sont pas montrés dans le texte converti
  - Ce n'est pas un bloc : chaque ligne de commentaire doit commencer par "%"
  - Pratiques pour les TODO, rappels FIXME et les notes de l'éditeur

Gras, italique et souligné

- **Description :** Utilisé pour insérer du texte gras/italique/souligné dans un paragraphe, une table, une liste ou une citation
- **Propriétés :** !Multiline, !FreeSpaces, !Align, Nesting

- **Contenu :** Macros, Beautifiers, Raw, Links, Image
- **Syntaxe :**
  - Deux étoiles autour pour le gras `**comme ceci**`
  - Deux barres autour pour l'italique `//comme ceci//`
  - Deux signes soulignés pour le souligné `__comme ceci__`
  - Les marques doivent collées au contenu (pas d'espace) `** ceci ** est invalide`
- **Détails :**
  - Tous les modificateurs de texte doivent être sur une seule ligne du fichier source, il ne doit pas y avoir de coupure de ligne
  - Les macros dont autorisées à l'intérieur : `**%%date**`
  - Vous pouvez mixer les modificateurs à l'intérieur `**__comme__ //ceci//**`

fonte non proportionnelle

- **Description :** Utilisé pour mettre un texte en fonte non proportionnelle dans un paragraphe, un table, une liste ou une citation.
- **Propriétés :** !Multiline, !FreeSpaces, !Align, !Nesting
- **Contenu :** −
- **Syntaxe :**
  - Deux apostrophes inverses ``` ``comme ceci`` ```
  - Les marques doivent être collées au contenu (pas d'espace) ``` `` ceci `` est invalide ```
- **Détails :**
  - Les marques ne sont pas interprétées
  - Les macros ne sont pas interprétées
  - Tout le texte en fonte non proportionnelle doit être sur une seule ligne, pas de coupure de ligne dans le contenu
  - Dans quelques cibles les espaces internes sont maintenus, dans d'autres les espaces multiples sont remplacés par un seul.
  - Vous pouvez créer un texte gras en fonte non proportionnelle en l'insérant dans les marques de gras : ``**``gras non proportionnel``**``. La même chose pour les autres modificateurs comme ``//``italique``//`` et ``__``souligné``__``.

ligne et zone Verbatim

- **Description :** Utilisé pour insérer du code programme ou autre texte pré−formatté, préservant les espaces et les sauts de ligne et en utilisant une fonte non proportionnelle
- **Propriétés :** Multiline, !FreeSpaces, !Align, !Nesting
- **Contenu :** −
- **Syntaxe : ligne Verbatim :**
  - Une ligne commençant avec exactement 3 apostrophes inverses suivis d'un espace, et suivi du texte ``` ``` comme ceci ```
  - Les apostrophes inverses doivent être en première colonne, sans espace devant
- **Syntaxe : Zone Verbatim :**
  - Une ligne commençant avec exactement 3 apostrophes inverses ``` ``` ``` suivie par les lignes de texte, suivis d'une autre ligne avec exactement 3 apostrophes inverses ``` ``` ```
  - **AUCUN** espace n'est autorisé avant et après les marques
- **Détails :**
  - Les marques ne sont pas interprétées
  - Les macros ne sont pas interprétées
  - Si on atteint la fin du fichier (EOF), la zone verbatim est terminée

Ligne de séparation, ligne épaisse

- **Description :** identifie une ligne de séparation ou une ligne épaisse
- **Propriétés :** !Multiline, FreeSpaces, !Align, !Nesting
- **Contenu :** −
- **Syntaxe :**
  - ♦ LA ligne de séparation peut tre composée de tirets "−" ou de caractères soulignés "_"
  - ♦ La ligne épaisse est composée par des signes égal "="
  - ♦ Il faut au moins 20 tirets/soulignés/egal
  - ♦ Des espaces optionnels peuvent être mis en début ou en fin de ligne
  - ♦ Tout autre caractère sur la ligne invalide la marque
- **Détails :**
  - ♦ Si la cible n'admet pas de ligne de séparation, une ligne de commentaire la remplace
  - ♦ La ligne épaisse peut avoir une traduction différente sur certaines cibles :
    - ◊ Une ligne de séparation plus épaisse
    - ◊ Une pause dans les formats de présentation comme Magic Point
    - ◊ Une coupure de page pour les cibles avec pagination comme LaTeX

Liens, liens nommés

- **Description :** identifie une lien distant (internet) ou local
- **Propriétés :** !Multiline, !FreeSpaces, !Align, !Nesting
- **Contenu :** Macros, Raw, Image
- **Syntaxe :**
  - ♦ Toute adresse internet valide URL, ftp, news ou adresse email est détectée et une adresse est convertie automatiquement
  - ♦ Le protocole (http, https, ftp) est optionnel `www.comme−ceci.com`
  - ♦ Un nom peut être utilisé pour un lien : `[cliquer ici www.url.com]`
  - ♦ Une image peut pointer sur un lien : `[[image.jpg] www.url.com]`
  - ♦ Les macros sont autorisées dans les adresses de lien : `[voir source %%infile]`
  - ♦ Les macros sont autorisées dans le nom du lien : `[mirror of %%outfile www.url.com]`
  - ♦ Le lien doit être spécifié sur une seule ligne, sans coupure de ligne
- **Détails :**
  - ♦ Si la cible n'a pas de support de liens elle sera soulignée

Quote

- **Description :** identifie une ligne citation indentée
- **Propriétés :** Multiline, !FreeSpaces, !Align, Nesting
- **Contenu :** Macros, Beautifiers, Quote, Raw, Bars, Links, Image, Comment
- **Syntaxe :**
  - ♦ Une ligne qui commence par une tabulation (TAB)
  - ♦ Plus il y a de tabulation plus la profondeur est grande
  - ♦ Les citations ne sont pas autorisées dans les listes et les tables
- **Détails :**
  - ♦ Si la fin du fichier est atteinte (EOF), la citation ouverte est terminée
  - ♦ Certaines cibles ne supportent pas différents niveaux de citation, alors les "sous−citations" sont transformées au niveau supérieur
  - ♦ Il n'y a pas de limitation pour la profondeur. Mais certines cibles peuvent avoir des restrictions, les sous−citations sont mises au niveau supérieur

Listes, listes numérotées, listes de définition

- **Description :** identifie le début d'une liste d'items
- **Propriétés :** Multiline, !FreeSpaces, !Align, Nesting
- **Contenu :** Macros, Beautifiers, Lists, Table, Verbatim, Raw, Bars, Links, Image, Comment
- **Syntaxe :**
    - ♦ Une ligne commence par un tiret/plus/deux points suivi d'un seul espace
- le premier caractère de la liste ne peut PAS être un espace (sauf dans les listes de définition)
    - ♦ Des espaces (pas des TAB) en début de ligne définissent la profondeur de l'indentation
    - ♦ Les sous−listes se terminent par une sous−liste de moindre profondeur ou avec un item vide
    - ♦ Toutes les listes ouvertes sont fermées par deux lignes blanches consécutives
- **Détails :**
    - ♦ Si la fin de fichier est atteinte (EOF), toutes les listes ouvertes sont terminées
    - ♦ Les listes peuvent être mixées, comme une liste de définition dans une liste numérotée
    - ♦ Quelques cibles n'admettent pas les imbrications de listes, les items des sous−listes sont transformées dans le niveau supérieur
    - ♦ Il n'y a pas de limite pour la profondeur des listes. Mais certaines cibles ont des restrictions, les sublistes de niveau inférieur sont alors remontées

Image

- **Description :** Identifie une image
- **Propriétés :** !Multiline, !FreeSpaces, Align, !Nesting
- **Contenu :** Macros
- **Syntaxe :**
    - ♦ Un nom de fichier image encadré entre deux crochets, `[comme_ceci.jpg]`
    - ♦ Le fichier doit se terminer par une extension comme PNG, JPG, GIF, ... (la casse est indifférente)
    - ♦ les symboles ne sont pas autorisés dans le nom de fichier `[comme_ceci!~1.jpg]`
    - ♦ Les macros sont autorisées dans le nom de fichier `[report−%%date(%Y−%m−%d).png]`
    - ♦ PAS d'espace dans le nom de fichier `[comme ceci.jpg]`
    - ♦ PAS d'espace entre le crochet et le nom de fichier `[ comme−ceci.jpg ]`
- **Détails :**
    - ♦ Si la cible n'a pas de support d'image, le nom de fichier est entre (parenthèses)
    - ♦ La position des marques indique l'alignement de l'image :
        - ◊ `[GAUCHE.jpg]` blablablabla
        - ◊ blablablabla `[CENTRE.jpg]` blablablabla
        - ◊ blablablabla `[DROITE.jpg]`

Table

- **Description :** Délimite une ligne d'un tableau avec n'importe quel nombre de colonnes
- **Propriétés :** Multiline, FreeSpaces, Align, !Nesting
- **Contenu :** Macros, Beautifiers, Raw, Links, Image, Comment
- **Syntaxe :**
    - ♦ Une barre verticale "|" identifie la ligne d'un tableau
    - ♦ Deux lignes verticales au début dela ligne "||" identifient la ligne de titre d'un tableau
    - ♦ les espaces avant le première barre verticale personnalisent le centrage de ta table
    - ♦ Les champs sont séparés par la chaine " | " (espace barre−verticale espace)
    - ♦ Une barre verticale "|" à la fin de la première ligne du tableau rend les bordures

visibles
    - ♦ La barre verticale "|" à la fin des autres colonnes de la table est ignoré
    - ♦ Les espaces dans chaque cellule spécifient leur alignement
    - ♦ Exemple: `| table | row | with | five | columns |`
- **Détails :**
    - ♦ Chaque ligne du tableau doit être sur une seule ligne du fichier source, sans saut de ligne
    - ♦ Les cibles avec alignement par colonne (comme sgml et LaTeX), utilisent l'alignement de la première ligne comme référence
    - ♦ Toute ligne "non table" la termine, sauf les lignes de commentaire
    - ♦ Le nombre de cellules est flexible, chaque ligne peut avoir un nombre différent de cellules
    - ♦ il n'y a pas de moyen de spécifier la largeur d'une colonne ou d'une rangée
    - ♦ Si la cible n'admet pas les tableaux, les lignes de la table sont considérés comme une zone Verbatim

caractères, lignes et zones tel−quel (raw)

- **Description :** Utilisée pour "protéger" du texte du fichier source, pour que les marques et les macros ne soient pas traitées.
- **Propriétés :** !Multiline, !FreeSpaces, !Align, !Nesting
- **Contenu :** −
- **Syntaxe : caractères tel−quel :**
    - ♦ trois apostrophes doubles autour `""comme ceci""`
    - ♦ Les marques sont collées au contenu (pas d'espace entre les marques et le contenu)
- **Syntaxe : ligne tel−quel :**
    - ♦ Une ligne commençant par trois apostrophes doubles `""" comme ceci`
    - ♦ La marque doit être au début de la ligne en première colonne
    - ♦ Utiliser un espace après les doubles apostrophes pour les s'eparer du texte.
- **Syntaxe : zone tel−quel :**
    - ♦ Une ligne avec exactement 3 doubles apostrophes suivis de la ligne de texte, suivi d'exactement 3 doubles apostrophe
    - ♦ PAS d'espace avant et après les marques
- **Détails :**
    - ♦ Les marques ne sont pas interprétées
    - ♦ Les macros ne sont pas interprétées
    - ♦ Si la fin du fichier (EOF) est atteinte la zone tel−quel (raw) est fermée.

Sixième partie − maitriser le macros

Les macros sont des mots clés spécifiques qui sont développées au moment de la conversion. Elles sont utilisées pour insérer des informations dynamiques comme la date ou des informations sur les documents source et converti.

Une macro est constituée par les caractères %% suivis par son nom comme dans `%%date`. Quelques macros acceptent une chaine de format entre parenthèses juste après le nom de la macro comme `%%date(%Y−%m−%d)`. Cette chaine de format inclut du texte classique avec des directives identifiées par le signe % suivi d'un caractère d'identification. Si aucune chaine de format n'est fournie, le format par défaut est utilisé.

Nom de la macro	Se développe en ...	Format par défaut
%%date	la date courante	%Y%m%d
%%mtime	la date de modification du fichier source	%Y%m%d
%%infile	le chemin du fichier source	%f
%%outfile	le chemin du fichier de sortie	%f
%%toc	la table des matières du document(TOC)	−

Règles générales :

- le nom de la macro est insensible à la casse : `%%date`, `%%DaTe` et %%DATE sont identiques
- le macros sont valides dans la zone ENTETE et la zone CORPS sauf `%%toc` valide seulement dans la zone CORPS
- Une macro démarre la zone CORPS si elle est trouvée dans la zone CONFIGURATION
- Une macro peut être n'importe où dans la ligne (sauf la macro `%%toc` qui doit être seule sur une ligne)
- Une macro peut être utilisée dans les liens et les marques d'image (sauf `%%toc`)
- Les macros ne sont pas traitées dans les titres, le verbatim et le texte brut

Exemple complet (les zones en gras montrent les macros développées) :

> Ceci est le manuel utilisateur de txt2tags, converti en **html** par txt2tags à partir du fichier source **userguide_fr.t2t**. La conversion a été faite le **2005−06−14 23:16:17**, mais la dernière modification dans le document source a été faite le **2005−03−18 23:44:05**. Les fichiers source et résultats sont dans le répertoire **userguide**.

%%date

La macro `%%date` se développe en date et heure courante. C'est très pratique pour les entêtes et les pieds de document, pour indiquer quand le document a été généré. Pour développer la dernière date de modification voir la macro %%mtime.

Cette macro accepte un certain nombre de directives de format. La liste complète peut être consultée sur le site Python.

Voici les plus couramment utilisées :

Directive	Description
%a	jour de la semaine en abrégé d'après la locale
%A	jour de la semaine d'après la locale

%b	nom du mois en abrégé d'après la locale
%B	nom du mois d'après la locale
%c	jour et heure d'après la locale
%d	jour du mois en chiffre [01,31]
%H	heure (24−heures) en chiffre [00,23]
%I	heure (12−heures) en chiffre [01,12]
%m	mois en chiffre [01,12].
%M	Minute en chiffre [00,59].
%p	équivalent d'après la locale de AM/PM
%S	iseconde en chiffre [00,61]. (1)
%x	représentation de la date d'après la locale
%X	représentation de l'heure d'après la locale
%y	année sans les siècles en chiffre [00,99].
%Y	année complète avec les siècles en chiffre
%%	caractère "%"

Exemples:

Macro	**−−>**	**Resultats pour la date 2005, Jun 14 at 23:16**
%%date(Converti le: %c)	−−>	Converti le : Tue Jun 14 23:16:17 2005
%%date(%Y−%m−%d)	−−>	2005−06−14
%%date(%I:%M %p)	−−>	11:16 PM
%%date(Aujourd'hui est le %A, du mois %B.)	−−>	Aujourd'hui est le Tuesday, du mois June.

%%mtime

La macro `%%mtime` se développe dans la date de dernière modification du fichier source. C'est utile pour noter quand le fichier a été changé la dernière fois. Cette macro est la "soeur" de la macro %%date, et accepte les mêmes chaines de format.

Comme exemple : ce guide utilisateur a été édité la dernière fois le **Fri Mar 18 23:44:05 2005**. Cette date est développée à partir de `%%mtime(%c)`.

%%infile

La macro `%%infile` se développe en chemin du fichier source dans le système. C'est utile pour dire "*voir le souirce de ce fichier*" sur les liens des pages HTML. Donner cette information est une attitude aimable avec les débutants, pour qu'ils puissent utiliser votre source comme exemple pour leurs propres pages.

Cette macro accepte les chaines de format suivantes :

%<caractère>	**Description**	**Sortie pour ce manuel utilisateur**
%f	nom du fichier	userguide_fr.t2t

%F	nom du fichier (sans extension)	userguide_fr
%e	extension du fichier	t2t
%p	chemin absolu du fichier	/a/txt2tags/src/doc/fr/userguide/userguide_fr.t2t
%d	chemin du fichier (répertoire seulement)	/a/txt2tags/src/doc/fr/userguide
%D	chemin du fichier (répertoire parent seulement)	userguide
%%	caractère "%" seul	%

Exemples:

Source	−−>	Développement
Ce guide est dans le répertoire %%infile(%D).	−−>	Ce guide est dans le répertoire userguide.
j'utilise l'extension %%infile(%e)	−−>	j'utilise l'extension t2t
[voir le source %%infile]	−−>	voir le source
Convertit en XHTML, cela donne %%infile(%F).xhtml	−−>	Convertit en XHTML, cela donne userguide_fr.xhtml

Note: La macro et développée en "−" si la source vient de STDIN

%%outfile

La macro `%%outfile` se développe dans le nom du fichier converti dans le système. C'est pratique dans les zones CORPS et ENTETE. Cette macro est une "soeur" de la macro %%infile et accepte les mêmes chaines de format.

Exemples:

Source	−−>	Développement
vous lisez le fichier %%outfile	−−>	vous lisez le fichier userguide_fr.html
txt2tags −t %%outfile(%e) −i %%infile −o %%outfile	−−>	txt2tags −t html −i userguide_fr.t2t −o userguide_fr.html

Note: La macro et développée en "−" si la sortie est STDOUT

%%toc

La macro `%%toc` se développe en table des matières du document. C'est pratique pour indiquer où vous voulez l'insérer. Vous pouvez l'utiliser plusieurs fois et la placer à la fin du document. Ce guide utilise %%toc pour positionner la table des matières.

Différente des autres macros, cette macro n'accepte pas de chaine de format et a ses propres règles :

- valide que dans le CORPS du document
- doit être seule sur la ligne (des espaces avant et après sont autorisés)
- doit être utilisée avec l'option en ligne de commande −−toc sinon elle sera ignorée
- le positionnement et le format par défaut sont dévalidés quand la macro %%toc est trouvée

Septième partie − maitriser les réglages

Ces réglages sont des réglages spéciaux placés dans la zone CONFIGURATION du document source qui modifient le comportement de la conversion. Ces réglages sont tous optionnels. Les utilisateurs moyens peuvent les ignorer. Mais ils créent une dépendance, si vous commencez à les utiliser, vous ve pourrez plus vous en passer.

Le lignes de réglage sont des *lignes de commentaire spéciales* précédées d'une marque ("!") qui les rendent différents des commentaires classiques. La syntaxe est aussi simple que le réglage des variables, composés d'une clé et d'une valeur séparés par deux points (":").

%! clé : valeur

Détail de la syntaxe :

- le point d'exclamation doit être collé avec le caractère commentaire sans espace ("%!")
- les espaces entre la clé et le séparateur (":") sont optionnels
- le mot clé et la valeur sont insensibles à la casse (pas de distinction majuscule−minuscule)

Règles:

- Ces réglages ne sont valides que dans la zone CONFIGURATION et sont pris comme des commentaires dans la zone CORPS du document
- Si la même clé apparait plusieurs fois dans la zone CONFIGURATION, la dernière occurence sera celle utilisée. Exception : `options, preproc et postproc` sont cumulatifs.
- Une ligne de réglage avec une clé non valide est considérée comme du commentaire
- ces réglages ont priorité sur le fichier RC mais pas sur les options en ligne de commande

%!Target

En utilisant le réglage `target` un format cible par défaut est défini pour le document

```
%!target: html
```

Ainsi il sufit de lancer la conversion par :

```
$ txt2tags file.t2t
```

Et la conversion sera faite dans la cible spécifiée

Le réglage `target` ne supporte pas de spécification d'option de cible. La déclaration `%!target(tex): html` n'a pas de sens ...

%!Options

Ecrire une longue ligne de commande avec plein d'options n'est pas agréable et source d'erreur. Les réglages `options` permettent à l'utilisateur de sauver les options de conversion dans le document source. Cela garantit que le document source sera toujours converti de façon identique avec les mêmes options.

Il suffit d'écrire les options sans erreur de syntaxe comme vous le feriez dans la ligne de commande sans mettre l'appel du programme txt2tags, la spécification de cible et le nom du fichier source.

Par exemple si vous tapez cette ligne de commande pour convertir votre document :

```
$ txt2tags −t html −−toc −−toc−level 2 −−enum−title file.t2t
```

Vous vous simplifiez la vie en tapant dans la zone CONFIGURATION les options suivantes dans le document source :

```
%!target: html
%!options(html): −−toc −−toc−level 2 −−enum−title
```

Et en ligne de commande vous tapez "`txt2tags file.t2t`", et vous pouvez lancer directement la conversion à partir de votre éditeur favori par exemple pour vi :

```
:!txt2tags %
```

%!Encoding

Ce réglage est nécessaire aux non anglophones, qui utilisent des caractères accentués ou d'autres détails spécifiques à la locale. Ainsi l'encodage de la cible peut être personnalisé.

Les valeurs valides pour le réglage de l'encodage sont les mêmes que pour les document HTML comme *iso−8859−1* et *koi8−r*. Si vous n'êtes pas sur consultez cette adresse.

La cible LaTeX utilise des noms d'alias pour l'encodage. Ce n'est pas un problème pour l'utilisateur car txt2tags traduit les noms de façon interne. Quelques exemples :

txt2tags/HTML	**>**	**LaTeX**
windows−1250	>>>	cp1250
windows−1252	>>>	cp1252
ibm850	>>>	cp850
ibm852	>>>	cp852
iso−8859−1	>>>	latin1
iso−8859−2	>>>	latin2
koi8−r	>>>	koi8−r

Si la valeur est inconnue elle est transmise telle quelle autorisant à l'utilisateur des encodages spécifiques.

%!PreProc

Le réglage PreProc est un filtre d'entrée utilisé sur le document source. C'est une ressource "trouve et remplace" appliquée après que la ligne a été lue du document source et avant d'êtres traitée par txt2tags.

C'est utile pour définir des abréviations pour du texte tapé de façon répétitive comme :

```
%!preproc JJS          "John J. Smith"
%!preproc RELEASE_DATE "2003−05−01"
%!preproc BULLET       "[images/tiny/bullet_blue.png]"
```

Ainsi l'utilisateur peut écrire une ligne comme :

```
Bonjour, je suis JJB. Aujourd'hui nous sommes le RELEASE_DATE.
```

Et txt2tags va "voir" la ligne suivante :

```
Bonjour, je suis John J. Smith. Aujourd'hui nous somes le 2003−05−01.
```

Ce filtre agit entre l'auteur du document et la conversion txt2tags. C'est une première conversion avant la "vraie" conversion. Il se comporte comme un filtre Sed/Perl externe appelé de cette façon :

```
$ cat file.t2t | preproc−script.sh | txt2tags −
```

Le traitement de txt2tags commencera après que les substitutions PreProc aient été appliquées.

%!PostProc

Le régalge PostProc est un filtre de sortie appliqué sur le document converti. C'est une ressource "trouve et remplace" appliquée après le traitement de txt2tags.

Il est utile pour faire quelques finitions sur le document de sortie, changer des marques ou ajouter un texte particulier. Des exemples :

```
%!postproc(html): '<BODY.*?>' '<BODY BGCOLOR="green">'
%!postproc(tex) : "\\newpage" ""
```

Ces filtres changent la couleur de fond des pages HTML et enlèvent les sauts de pagfe en LaTeX.

Les règles PostProc sont comme un filtre Sed/Perl externe appelé de cette façon :

```
$ txt2tags −t html −o− file.t2t | postproc−script.sh > file.html
```

Avant que cette fonctionnalité ne soit intégrée, il était courant d'avoir des petits scripts pour ajuster le résultat de txt2tags. Ces scripts étaient en fait des commandes `sed`, pour faire des remplacements. Maintenent ces chaines de remplacement peuvent être sauvées avec le document source, et le plus est l'utilisation possible des expressions régulières pour trouver les motifs.

%!Style

- Utile pour les cibles HTML et XHTML, il définit un fichier CSS pour le document résultat.
- utile pour la cible LaTeX pour charger des modules par `\usepackage`.
- le même résultat est obtenu par l'option en ligne de commande `−−style`.
- l'option −−style est plus forte que le réglage %!style. Si les deux sont utilisés l'option −−style sera appliquée.

Définir un réglage pour une cible spécifique

Tous les réglages (sauf %!target) peuvent être liés avec une cible spécifique en utilisant la syntaxe : `%!clé(target): valeur`. De cette façon vous pouvez définir plusieurs configurations pour différentes cibles.

Cela est particulièrement utile pour les filtres pre/postproc, mais est applicable à tous les réglages. Par exemple définir des styles différents pour HTML et LaTeX :

```
%!style(html): fancy.css
%!style(tex) : amssymb
```

Cela est également utile pour ajuster les options sur le document converti :

```
%!target: sgml
%!options(sgml): --toc
%!options(html): --style foo.css
%!options(txt ): --toc-only --toc-level 2
```

Dans cet exemple, la cible par défaut est sgml et génèrera une table des matières. Si on lance la commande : `txt2tags -t html file.t2t` seules les options html seront utilisées, ainsi le fichier converti utilisera la feuille de style "foo.css" et n'aura pas de table des matières.

Details sur les filtres PreProc et PostProc

- Les filtres sont des fonctions "trouve et remplace" (pensez Sed)
- les filtres ne sont pas "utilise le dernier trouvé", ils sont cumulatifs. Vous pouvez définir sans limite autant de filtres que vous avez besoin. Ils seront appliqués dans l'ordre dans lequel ils sont définis.
- différents des autres réglages, les filtres spécifiques pour une cible et ceux génériques sont appliqués. s'appliquant sur toutes les cibles. Dans l'exemple suivant les deux filtres sont utilisés sur une cible HTML.

```
%!postproc      :   this   that
%!postproc(html):   that   other
```

- Les filtres doivent avoir deux arguments
- les séquences spéciales comme `\n` (saut de ligne) et `\t` (tabulation) sont interprétés
- pour supprimer du texte utilisez une chaine vide

```
%!postproc: "chaine bnon désirée" ""
```

- pour éviter les problèmes, spécifiez toujours la cible quand vous utlisez PostProc pour changer les marques : `%!PostProc(target): <ceci> <celat>`
- PreProc est appliqué après que la ligne a été lue et PostProc est appliqué après que la conversion ait été faite. Attention à l'UUOC (Useless Use Of Cat : utilisation inutile de cat) :

```
$ cat file.t2t | preproc.sh | txt2tags | postproc.sh
```

- la première partie du filtre (la chaine "cherche") n'est pas lue comme une chaine mais comme une expression régulière. Si vous ne savez pas ce que c'est, ne vous en souciez pas, vous pouvez ne jamais en avoir besoin. Mais gardez en mémoire que vous devez "échapper" des caractères pour l'utiliser. Echapper veut dire faire précéder la caractère par "\". En voici la liste :

```
\*  \+  \.  \^  \$  \?  \(  \)  \{  \[  \|  \\
```

- les experssion régulières Python sont disponibles. Elles sont identiques aux Perl Regexes (PCRE). Exemple: Changer les marques en gras "<B>" et"</B>" en "STRONG" en HTML :

```
%!postproc(html):   '(</?)B>'   '\1STRONG>'
```

- les arguments des filtres peuvent être passés de 3 façons :
    1. une simple chaine comme FOO (sans espace)
    2. une chaine avec des apostrophes doubles comme "FOO"
    3. une chaine avec des apostrophes comme 'FOO'
- Si votre motif a des apostrophes doubles, protégez le avec des apostrophes simples et vice−versa. Quelques exemples valides :

```
%!postproc:   PATT    REPLACEMENT
%!postproc:  "PATT"  "REPLACEMENT"
%!postproc:  'PATT'  'REPLACEMENT'
%!postproc:   PATT   "REPLACEMENT"
%!postproc:  "PATT"  'REPLACEMENT'
```

Hitième partie − magie noire

Ce chapitre n'est pas recommandé aux débutants. Il montre comment faire des choses étranges avec les filtres txt2tags, en utilisant des expressions régulières et des motifs complexes.

> **ATTENTION** : les manipulations suivantes ne sont pas encouragées et peuvent casser des choses. Un partie du texte de sortie peut être perdu lors de la conversion, n'apparaissant pas dans le document de sortie. N'utilisez ces trucs que si vous en avez réellement besoin et que savez ce que vous faites.

Les filtres sont une fontionnalité puissante mais dangereuse

De mauvais filtres génèrent des résultats inattendus.

SVP ne l'oubliez pas.

Insérer plusieurs lignes avec %!PostProc (comme des règles CSS)

Dans les filtres, le motif de remplacement peut inclure plusieurs lignes en utilisant la chaine de coupure de ligne `\n`.

Elles peuvent être utiles pour inclure de courtes règles CSS dans une cible HTML, supprimant le besoin d'un fichier supplémentaire.

```
%!postproc: <HEAD>      '<HEAD>\n<STYLE TYPE="text/css">\n</STYLE>'
%!postproc: (</STYLE>)  'body     { margin:3em               ;} \n\1'
%!postproc: (</STYLE>)  'a        { text-decoration:none     ;} \n\1'
%!postproc: (</STYLE>)  'pre,code { background-color:#ffffcc ;} \n\1'
%!postproc: (</STYLE>)  'th       { background-color:yellow  ;} \n\1'
```

Tous ces filtres sont liés au premier, en remplaçant une chaine qui a été insérée. Ainsi un simple "<HEAD>" se change en :

```
<HEAD>
<STYLE TYPE="text/css">
body     { margin:3em               ;}
a        { text-decoration:none     ;}
pre,code { background-color:#ffffcc ;}
th       { background-color:yellow  ;}
</STYLE>
```

créer un contenu spécifique à la cible avec %!PostProc

Quelquefois vous devez inclure du texte dans une seule cible, mais pas dans les autres. Ce comportement peut être obtenu en utilisant des astuces de PreProc.

L'idée est d'inclure un texte spécial en commentaire dans le document source, mais de le marquer de telle façon qu'un filtre spécifique à une cible le décommente.

Par exemple si un paragraphe doit être ajouté uniquement dans la cible HTML comme cela :

```
%html% Cette page HTML est générée par [txt2tags http://txt2tags.sf.net].
%html% voir le fichier texte source [ici source.t2t].
```

Comme ces lignes commencent par % ce sont des lignes de commentaire et seront ignorées. En ajoutant ce filtre :

```
%preproc(html): '^%html% ' ''
```

La chaine "%html% est supprimée et ces lignes deviennent visibles, n'étant plus des commentaires. Comme une cible est spécifiée ce filtre ne sera actif que pour générer du html.

Changer les marques de txt2tags avec %!PreProc

Si l'utilisateur est un spécialiste en expressions régulières, il peut personnaliser la syntaxe du document source en changeant les marques par défaut en quelque chose de plus "confortable".

Par exemple une tabulation en début de ligne est la marque de la citation. Si cela ne plait pas à l'utilisateur, ou que son éditeur a un comportement bizarre avec les tabulations, il peut définir une nouvelle marque pour les citations. Par exemple ">>>". Il ajoute ce simple iltre :

```
%!PreProc: '>>> ' '\t'
```

Et dans le document source, la citation sera indiquée par :

```
>>> Ceci est une citation.
>>> L'utilisateur a défini cette marque.
>>> Mais elles seront converties en tabulations par PreProc.
```

Avant que la conversion ne commence, les marques ">>>" seront converties en tabulations et txt2tags reconnaitra du les citations

> **ATTENTION** : des règles PreProc peuvent changer toute la syntaxe des marques, générant éventuellement des conflits entre les marques. Soyez très attentif quand vous faites cela.

The End.