

tpl4md : template markdown pour la génération de documents

Objet

tpl4md fournit des templates en markdown pour des documents largement utilisés comme des documents en pdf simple, des documents pdf complexes, des lettres, des factures, des bons de commande, ou même des slides. Le but est de pouvoir de concentrer sur le contenu qui sera écrit en markdown et de ne pas s'occuper des problématiques de mise en page pris en comptes par les templates. Le reste est en effet géré par pandoc, LaTeX, etc.

Traduction

Ce fichier README est aussi disponible [en anglais](#)

Pré-requis

Linux/Mac OSX

Le projet se sert des différentes technologies suivantes:

- Le logiciel [Pandoc](#) disponible sur la plupart des plate-formes existantes. La version minimum nécessaire est la 1.9.4.2.
- Le système d'écriture de documents [LaTeX](#). Ne vous inquiétez pas, nous n'écrirons pas de LaTeX! Enfin un tout petit peu peut-être (les paquets `texlive`, `texlive-latex-recommended`, `texlive-xetex`, `ttf-bitstream-vera` et `texlive-latex-extra` sont suffisant)
- Le format [Markdown](#) pour l'écriture du document avec les extensions fournies par Pandoc dont on peut trouver la description sur [la page dédiée](#)
- L'outil de compilation [Make](#)
- Un éditeur de texte pour taper le markdown (sur Mac OSX il y a [Mou](#) qui n'est pas trop mal car il propose un peu de WYSIWYG)
- Une version récente de Python pour les scripts de configuration et de génération de template avec `python-tk` activé

Windows

- Le logiciel [Pandoc](#) disponible sur la plupart des plate-formes existantes. La version minimum nécessaire est la 1.9.4.2.

- Le système d'écriture de documents [LaTeX](#) avec les packages docbook pour les parties de template et les polices DejaVu. Ne vous inquiétez pas, nous n'écrirons pas de LaTeX! Enfin un tout petit peu peut-être. Sous Windows je conseille TexLive (l'essai a été concluant) ou alors MikTeX qui est la distribution conseillée par John Mac Farlane (le créateur de Pandoc) sur son site.
- Une version récente de Python (celle de la communauté ou celle d'ActiveState)
- Le format [Markdown](#) pour l'écriture du document avec les extensions fournies par Pandoc dont on peut trouver la description sur [la page dédiée](#)
- Un éditeur de texte pour taper le markdown (Notepad ++ semble aller, à voir s'il existe un éditeur WYSIWIG sympa)
- Les logiciels make/sed et l'ensemble des coreutils de gnu portés sur Windows
- Les polices d'écriture présentes dans le svn (Marketing/03-Print/fonts) ajoutées au système d'exploitation

Marche à suivre pour l'installation sous Windows

- Installation de Pandoc : se rendre sur la page de download et télécharger le fichier **msi** : <https://code.google.com/p/pandoc/downloads/list>
- Installation des outils unix nécessaires dans un répertoire facile d'accès comme C:\gnuwin32 :
- Make : <http://gnuwin32.sourceforge.net/downlinks/make.php>
- Sed : <http://sourceforge.net/projects/gnuwin32/files/sed/4.2.1/sed-4.2.1-setup.exe/download>
- CoreUtils : <http://gnuwin32.sourceforge.net/downlinks/coreutils.php>
- Installation de Python : celle-ci <http://python.org/ftp/python/2.7.5/python-2.7.5.msi> ou celle-ci <http://www.activestate.com/activepython/downloads/thank-you?dl=http://downloads.activestate.com/ActivePython/releases/2.7.2.5/ActivePython-2.7.2.5-win32-x86.msi>
- Installer la distribution LaTeX TexLive <http://mirror.ctan.org/systems/texlive/tlnet/install-tl.zip> qu'il faut décompresser pour ensuite exécuter le logiciel *install-tl-advanced.bat* qui vous permettra d'installer l'ensemble des paquets LaTeX. Il est bien important de connaître le chemin où elles ont été installées : C:\texlive2012 par exemple.

Une fois tout ces logiciels installés ou téléchargés, il reste quelques opérations à effectuer :

- Ajouter les chemins des dossiers binaires contenant les exécutables des logiciels correspondants dans les variables PATH (globale) et Path (spécifique par utilisateur). Ces variables sont modifiables dans le panneau de configuration, en cliquant sur "**Systèmes -> Configuration le nom de cet ordinateur -> dans la colonne à gauche Paramètres avancés**".

Une fenêtre apparaît avec un bouton variables d'environnement sur lequel il faut cliquer. Dans la fenêtre vous avez :

- Sur la partie haute la liste des variables globales : il faut donc cliquer sur `PATH`, cliquer sur modifier et ajouter ensuite à ce qui est déjà entré la liste des chemins des répertoires que vous avez installé avec ajouté `bin` pour tous sauf python et il faut les séparer avec des `;`. Cela devrait donc donner quelque chose ça `...;C:\python27;C:\texlive2012\bin;C:\gnuwin32\bin`
- Sur la partie basse on retrouve les variables d'environnement par utilisation, il faut ajouter dans la variable `Path` la même chose.

Comment installer `tpl4md`

Depuis le repository github

Pour install `tpl4md` depuis le repository github :

```
$ cd output_path
$ git clone git@github.com:dloureiro/tpl4md.git .
```

Vous n'avez ensuite plus qu'à ajouter le dossier `bin` à votre variable `PATH`.

En utilisant `pip`

Pour installer `tpl4md` en utilisant l'installateur `pip` :

```
pip install tpl4md
```

`tpl4md` sera automatiquement ajouté à votre variable `PATH`.

Contenu du projet

Liste des fichiers

```
root
|- bin
|  |- marksf_gen
|  |- marksf_gen.bat
|- share
|  |- common
|  |  |- bin
```

```

| | | |- config_parsing
| | | |- generate_files
| | |- doc
| | | |- README.pdf
| | | |- README-fr.pdf
| | |- templates // La liste des templates disponibles
| | | |- epub // Jetez un coup d'oeil au README.md de chacun de ces templates
| | | |- pdf-cplx
| | | |- pdf-lettre
| | | |- pdf-simple
| | | |- invoice-fr
| | | |- order-fr
| | | |- slides-reveal
|- README.md
|- README-fr.md

```

Description de ce contenu

- **bin/tpl4md** : Générateur bostrappant un projet en fonction du template demandé
- **bin/tpl4md.bat** : Fichier batch pour le lancement du générateur pour windows
- **README(-fr).md** : Ce fichier de README et sa traduction en anglais
- **share/doc/README(-fr).pdf** : Ce fichier de README et sa traduction en anglais au format PDF
- **share/common/bin/config_parser** : Script permettant la récupération de certaines variables du fichier de configuration
- **share/templates** : Le dossier contenant les templates disponibles
- **share/templates/epub** : Dossier du template pour les ebooks en ePub
- **share/templates/pdf-simple** : Dossier du template pour les documents PDF simples
- **share/templates/pdf-cplx** : Dossier du template pour les documents PDF complexes
- **share/templates/pdf-lettre** : Dossier du template pour les lettres et courriers divers
- **share/templates/invoice-fr** : Dossier du template pour les factures
- **share/templates/order-fr** : Dossier du template pour les bons de commande
- **share/templates/slides-reveal** : Dossier du template pour des slides propulsées par [reveal.js](#)

Nous allons aborder les fichiers les plus importants et nous nous attarderons ensuite sur les templates fournis.

tpl4md

Ce script est l'élément principal du projet car il permet la génération d'un projet vierge pour la mise en place de document en suivant l'un des templates fournis. Ce script possède la signature suivante :

```
tpl4md [-h] [-l] [-o OUTPUT] [-i] [-t TYPE]
```

- **-l** ou **-list** : liste les templates disponibles (pour l'instant epub/pdf-simple/pdf-cplx/pdf-lettre)
- **-t** ou **-type** : définit le type de template à utiliser (l'un de ceux fournis par l'option **-l**)
- **-o** ou **-output** : définit le nom du répertoire (qui ne doit pas exister) et qui contiendra l'ensemble des éléments du projet généré
- **-i** ou **-interface** : permet d'avoir une suite de boîte de dialogue qui vont vous permettre de définir notamment :
 - le type de template à utiliser
 - le nom du dossier à créer
 - le répertoire dans lequel ce dossier va être créé

tpl4md.bat

Ce fichier est un fichier batch pour windows effectuant l'appel suivant : **tpl4md -i**

config_parser

Ce script permet la récupération de variables au sein du fichier de configuration. Il est principalement utilisé par le Makefile pour récupérer quelques informations de base comme le nom du fichier source ou le nom du fichier de destination.

Voici la façon de l'appeler :

```
python ./bin/config_parser key
```

avec *key* la clef dont on veut récupérer la valeur.

**** Attention : vous n'avez pas de raison particulière de l'appeler en dehors du Makefile. Il ne sert en effet qu'à construire un fichier temporaire ****

Template

Description principale

Un template reste assez libre dans son implémentation. Quelques règles sont néanmoins observées pour leur utilisation depuis le script de génération `tpl4md`

Nous allons ici aborder les grands principes d'un template. Vous pourrez ensuite vous référer à la documentation spécifique de chacun de ces templates pour plus de détail.

Chaque template est actuellement constitué des éléments de base suivant:

- Un Makefile possédant deux targets :
- la principale qui sert à la compilation
- `clean` qui sert au nettoyage des fichiers compilés

La target principale va être variable pour chaque template (la standardisation est inscrite dans les **TODO**) et est fourni avec les templates existants des fichiers batch qui vont appeler les tâches définies dans le **Makefile** pour les utilisations de windows. * Un fichier de configuration nommé `configuration.json` qui contiendra les informations de configuration pour l'adaptation du template choisi
* Un fichier `generator.py` qui va s'occuper de la copie des fichiers de base d'un template vers le dossier de destination lors de la génération d'un projet basé sur un template

Ensuite, chaque template contiendra un ensemble d'éléments qui peuvent lui être spécifique.

Templates globaux et spécifiques à l'utilisateur

`tpl4md` va chercher les templates disponibles dans deux dossiers :

- `share/templates` au même niveau que le dossier contenant les scripts de `tpl4md` et qui contiendra les templates globaux fournis avec la distribution
- `$HOME/.tpl4md/tempaltes` dans le dossier utilisateur qui contiendra les templates que l'utilisateur souhaitera rendre disponible à `tpl4md` qu'il aura pu développer lui-même ou trouver sur internet.

README.pdf

Le readme en pdf est générée avec la commande suivante :

```
pandoc -f markdown-raw_tex README.md -o README.pdf
```

TODO

Un certain nombre d'améliorations peuvent être apportées aux éléments déjà mis en place. En voici une liste non exhaustive:

- Standardisation des templates notamment pour la copie des fichiers d'un template
- Factorisation des éléments des templates
- Génération d'un document HTML
- **DONE** Ajouter la présentation Reveal.js comme template de document
- Standardisation des targets des makefiles
- **DONE**: Passage de `pdflatex` à `XeTeX` pour un meilleur support de l'UTF-8 mais aussi pour un meilleur support des polices non LaTeX
- amélioration des templates pdf simple et complexe pour que le watermark s'affiche au-dessus de l'image de fond, comme dans le template de lettre
- Écriture d'un guide pour la création d'un template
- **DONE**: Découpage du fichier contenant l'adaptateur de template basé sur le fichier `configuration.json`
- Ajout d'une feuille de style CSS ainsi qu'une cover pour le document epub
- **DONE**: Définition d'un template de document non-technique
- **DONE**: Pré-processeur qui prendrait en entrée un fichier au format JSON par exemple qui permettrait de ne pas avoir à modifier les templates quand on souhaite rédiger un nouveau document