Livre complexe

(présentant de nombreux cas d'impressions.)

Philippe Perret

Icare Éditions

Complexité testée

Présentation

Ce livre doit permettre de tester énormément de choses dans le détail. Grâce à l'affichage (par la recette) de la grille de référence, on peut voir dans le livre produit le résultat. Il va y avoir beaucoup de changements de fontes et de tailles (elles sont toutes définies dans le fichier recette).

Premiers tests

Ce texte est écrit dans la police “Helvetica” en taille 21 et style italique (c’est une ligne de pfbcode qui le détermine juste au-dessus). Elle doit juste s’adapter aux lignes de références qui sont placées dans cette partie (par défaut) avec une hauteur de ligne de 24 points.

On poursuit avec un texte en police “Times-Roman” en taille 12 et style romain. De la même manière, ce texte est assez long pour pouvoir se placer sur plusieurs lignes, afin de voir si le traitement par ligne fonctionne

correctement et que le texte se place bien et naturellement sur les lignes de référence tracées.

Un paragraphe dans la police normale, mais avec des *textes en italiques*, des **textes en gras** et des textes soulignés. Il possède aussi un 1er exposant et une 1re ré-utilisation d'exposant ainsi qu'une note [1] qui doit être placée en dessous du paragraphe, dans un style un peu différent.

[1] C'est le commentaire de la note qui a été placée plus haut, qu'on doit correctement formater en fonction des choix dans le livre de recette {{Cette définition doit être ajoutée}}

Tests restant à faire

* Traitement du formatage pseudo-markdown
* Traitement des veuves
* Traitement des orphelines
* Traitement des lignes de voleur
* La modification du THIEF_LINE_WIDTH à la volée, pour pouvoir modifier localement la longueur d'une ligne de voleur. On doit pouvoir aussi le faire dans la recette (autre test ?)
* Penser à ajouter la définition de l'aspect des notes dans la recette (manuel). Ajouter les valeurs par défaut dans RECIPE DEFAULT.
* Ajouter la page d'infos, avec toutes les infos (l'idée est qu'il y ait plus d'infos que de lignes, pour voir comment on va s'y prendre => doubler les lignes et mettre les caractères plus petits pour s'adapter.

Bogues à corriger / implémenter

* Les notes doivent être placées plus haut

* Les notes ont des paragraphes numérotés. Ne le faire que si l'option est demandée

Réflexions

Gestion des lignes complexes

Comment gérer les lignes complexes (les lignes qui ne sont pas simples…), à commencer par les items de liste. Mais on peut imaginer que les tables en sont aussi. Il faut vraiment parvenir à “sortir” le traitement par ligne. Par exemple en ayant un constructeur (Printer ?) à qui on envoie :

* un texte,
* peut-être une fonte générale

* des paramètres définissant la largeur :width et la position :left,

* le pdf (Prawn::document)

* un hauteur (cursor)

et qui calcule et imprime le texte. L'essai sera concluant si on parvient à imprimer une table (mais une fausse table) avec quatre colonnes de tailles différentes par exemple dont les cellules contiennent des textes assez longs, avec des fontes différentes, et des lignes (des lignes ajoutées par dessin, donc, puisque qu'on n'utiliserait pas les vraies).

Il faudrait cependant que les vraies tables puissent être utilisées, qui zapperaient l'alignement sur la grille de référence [Peut-être que l'utilisation actuelle fonctionnerait].

Traitement per paragraphe suivant

Dans le nouveau système, où les lignes sont directement injectées dans le livre, on ne peut plus faire de traitement en fonction de la ligne suivant. Typiquement, si on a des notes, on ne peut plus ajouter la ligne de fin à la dernière note s'il n'y a plus de notes après.

Deux solutins sont possibles :

* Indiquer qu'un bloc de note est en route (dans l'injecteur `Injector` qu'on pour instancier) et, lorsqu'on rencontre un paragraphe qui n'est plus une note, le fermer. Inconvénient(s) : ça oblige à suivre des trucs tout le temps (if block_notes? ....)

* Pour lire le paragraphe suivant. Pas si simple, même pour les fichiers (qui sont lus avec la méthode `readlines` (qui pourrait être remplacées par `gets`) et encore moins évident avec les méthodes utilisateurs.