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 sur première [1] qui doit être placée en dessous du paragraphe, dans un style un peu différent et une autre note sur deuxième [2].

[1] *C'est le commentaire de la note sur "première" qui a été placée plus haut, qu'on doit correctement formater en fonction des choix dans le livre de recette.*

[2] *Et le commentaire sur la note "deuxième" qui doit être bien numérotée en deuxième justement.*

Un paragraphe juste en dessous de la note pour

voir s’il serait bien placé sous le trait et non pas dessus ce qui serait inélégant.

Paragraphes avec formatage personnalisé

- Un premier paragraphe de style “simplenote” qui se caractérise par une puce “pictophil” (lettre “n”) placée au bon endroit en décalant le texte qui sera toujours bien placé.
- Une autre simple note pour voir si les caractères qui ne s’affichent pas vont s’afficher cette fois avec une ligne inférieure (en fait le problème vient de `.otf`).
- Une note pour de la documentation.

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.

* Utiliser les § (caplock §) pour désigner le numéro du paragraphe dans la numérotation hybride.

Bogues à corriger / implémenter

– Le tiret des items de list doivent avoir la fonte du texte (voir le premier qui prend la fonte du titre)
– Plus grave, les tirets d'item de liste ne doivent être mis que lorsque l'on sait où va se retrouver le paragraphe. S'il doit passer à la ligne, le tiret doit lui aussi passer à la ligne
– La largeur de l'item de liste doit être amputée du décalage left (pour le moment, il est normalement réglé (avec `margin-left`, mais le programme n'en tient aucun compte)

Réflexions

Second tour

Pour le second tour, on n’a absolument pas besoin de repartir du texte. On va plutôt repartir de tous les paragraphes, et modifier les références puisque normalement c’est la seule chose qui nécessite ce second tour (on a un troisième tour avec les définitions du scénodico, mais c’est un tout autre problème qui sera résolu).

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 des “lettrines”

J’appelle “puce” le tiret ou la puce qu’on trouve au début d’une ligne, mais aussi tout autre caractère à placer devant une ligne de texte qui sera mise en retrait gauche, comme par exemple toutes les notes (notereal, simplenote, etc.) qu’on utilise pour l’analyse de film.

Ce traitement doit être proposé de base, avec une option particulière qu’on appellera :puce dans les options à envoyer à Printer.pretty_render