

Manuel de Prawn-for-book

*(le manuel autoproduit de l'application
de mise en forme professionnelle)*

Philippe Perret

Icare Éditions

AVANT-PROPOS	13
FONCTIONNALITÉS	17
Les Recettes	19
Recette du livre	19
Recette de la collection	19
Définition de la fonte par défaut du texte	20
Définition des fontes (embarquées)	21
Éditeur / Maison d'édition	23
Le Texte du livre	25
Les types de paragraphes	25
Texte au format pseudo markdown	25
Formatages du texte	25
Introduction à la stylisation en ligne	26
Définition des titres	27
Aides & Assistants	30
Manuel et autres aides	30
Correspondances linguistiques	31
Les « Snippets »	31
Messages d'erreurs et notices	33
Afficher grille de référence et marges	34
Exportation du texte	34
Pagination	35
Les types de pagination	35
Types de numérotation	35
Numérotation des paragraphes	38
Suspension de la pagination	44
Le Texte en détail	49
Stylisation en ligne	49
Indentation des paragraphes	52
Gestion des tirets conditionnels	53
Placement sur une ligne quelconque	55
Les puces	56

Format du livre et des pages	69
Définition des marges	69
Section en multi-colonnes	72
Le Mode Expert	75
Description	76
Indication du contexte d'erreur	76
Référence à un paragraphe	76
Injection	78
Évaluation du code ruby	78
Méthodes d'helpers en mode expert	81
Parsing du paragraphe	82
Bibliographies en mode expert	83
Mode Multi-colonnes	83
Formateurs	85
Modification de la hauteur de ligne	85

Avant-propos

Ce manuel présente toutes les fonctionnalités, à jour, de l'application « Prawn-for-book » (« Prawn pour les livres »), application dont la principale vocation est d'**obtenir un document PDF professionnel prêt pour l'imprimerie** à partir d'un simple fichier de texte (contenant le contenu du livre, le roman par exemple).

Ce manuel de 94 pages est auto-produit par « **Prawn-for-book** », c'est-à-dire qu'il est construit de façon *programmative* par l'application elle-même. Il en utilise toutes les fonctionnalités puisqu'il génère de façon automatisé les exemples et notamment les *helpers* de mise en forme, les références croisées ou les bibliographies. En ce sens, ce manuel sert donc aussi de test complet de l'application puisqu'une fonctionnalité qui ne fonctionnerait pas ici ne fonctionnerait pas non plus dans le livre produit.

Si vous êtes intéressé(e) de voir comment il est généré, vous pouvez consulter principalement le fichier markdown `texte.pfb.md`, le fichier ruby `prawn4book.rb` et le fichier recette yaml `recipe.yaml` qui le définissent, dans le dossier `Manuel/manuel_building` de l'application.

Fonctionnalités

Les Recettes

Recette du livre

La *recette du livre* est un fichier de nom `recipe.yaml` qui se trouve à la racine du dossier du livre. Son nom vient de « recipe » qui signifie *recette* en anglais et de `.yaml`, extension des fichiers au format simple YAML (cf. annexe/format_yaml (p.)).

Vous pouvez voir ci-dessous un extrait du fichier recette de ce manuel (qui a bien sûr été produit par **Prawn-For-Book**).

Extrait de fichier recette

```
---
#<book_data>
book_data:
  title:      "Manuel de Prawn-for-book"
  author:     "Philippe PERRET"
  version:    1.3
  isbn:       null
  # .\..
#</book_data>
#<book_format>
book_format:
  book:
    width:    '210cm'
    height:   '297cm'
    orientation: portrait
    format:   :pdf
# .\..
```

Recette de la collection

La *recette de la collection* est un fichier de nom `recipe_collection.yaml` qui se trouve à la racine du dossier d'une collection de livres. Son nom vient de « recipe » qui signifie *recette* en anglais et de `.yaml`, extension des fichiers au format simple YAML (cf. page annexe/format_yaml (p.)).

Vous pouvez trouver ci-dessous les données propres à une collection.

Si le fichier `recipe.yaml` ou `recipe_collection.yaml` contient...

```
---
# (données de la collection)
collection:
  name: "Nom de la collection"
  short_name: "Nom raccourci (pour les messages)"
#/collection
# .\..
```

Définition de la fonte par défaut du texte

Bien que l'application propose, clé en main, une fonte qui peut être utilisée pour imprimer un livre, on peut définir n'importe quelle fonte comme fonte par défaut, et même une fonte personnelle dont on aura au préalable acheté la licence (c'est presque toujours nécessaire si le livre doit être vendu).

SI la recette du livre contient...

```
---
# Définition des fontes
# -----
fonts:
  Reenie:
    normal: "assets/fontes/Reenie_Beanie/ReenieBeanie-Regular.ttf"
#
# Format du livre (texte, page, etc.)
# -----
book_format:
  text:
    font: "Reenie/normal/20/000077"
    line_height: 32
```

ET que le texte du livre contient...

Un texte avec la police par défaut. C'est celle définie dans la recette, avec les autres données. Vous pouvez par exemple remarquer la hauteur de ligne (`line_height`) particulièrement haute ici.

ALORS le livre contiendra...

un texte avec la police par défaut. C'est celle définie dans la recette, avec les autres données. Vous pouvez par exemple remarquer la hauteur de ligne (`line_height`) particulièrement haute ici.

1

Définition des fontes (embarquées)

Bien que l'application propose, clé en main, des fontes à utiliser pour imprimer son livre, on peut définir n'importe quelle fonte dont on posséderait la licence commerciale pour être utilisée pour tout élément du livre (texte, titres, etc.). Mais afin que l'imprimeur puisse s'en servir, il faut *embarquer* ces fontes dans le document PDF à destination de cet imprimeur.

Conseil concernant l'emplacement des fontes

Pour éviter tout problème — notamment quand on déplace le dossier du livre ou qu'on le transmet à quelqu'un —, il est plus prudent de faire un dossier `fontes` dans votre dossier de collection ou de livre et d'y dupliquer les polices que vous voulez utiliser — c'est-à-dire les fichiers `.ttf` et `.otf`.

Si vous avez à transmettre le dossier à un collaborateur ou autre, celui-ci ou celle-ci pourra imprimer correctement le livre, avec les fontes désirées, même s'il ne les possède pas sur son ordinateur.

Styles obligatoires pour les fontes

Pour pouvoir utiliser la mise en forme de base du texte, c'est-à-dire les italiques, les gras, les soulignés, vous devez impérativement définir ces styles de police.

Il faut comprendre qu'en impression professionnelle, il n'y a pas de *magie* au niveau des polices, si vous utilisez de l'italique par exemple, ***Prawn-For-Book*** utilise la forme *italic* de votre police, si vous utiliser de l'italique et du gras, ***Prawn-For-Book*** utilise la forme *italic_bold* de votre police. Ces *formes* doivent toutes être définies.

Définition des fontes dans la section :*fonts*

Vous définissez les fontes (polices) dans une section `fonts` (« fontes » en anglais) de votre recette. C'est une table YAML dont les clés seront les noms de vos polices, ceux que vous utiliserez pour définir les annexe/font_string (p.) de chaque élément de votre texte, à commencer par la fonte par défaut.

Imaginons que vous ayez mis le fichier `New-Time-regular.ttf` de la police « New-Times » dans un dossier `fontes` de votre dossier de collection (ou de livre). Pour y faire référence dans les fontes-strings que vous voulez utiliser, vous voulez l'appeler simplement `NTime` (pour avoir un font-string qui ressemble à `NTime/regular/12/333333`), alors, dans la recette, il vous suffit de mettre :

```
# Dans recipe_collection.yaml
---
fonts:
  NTime:
    regular: "fontes/New-Time-regular.ttf"
```

Comme vous allez utiliser aussi de l'italique, du gras, et de l'italique avec du gras, et que vous possédez (dans le dossier `polices` de votre ordinateur ou le dossier de la police dont vous venez d'acheter la licence) les fichiers relatifs qui sont `New-Time-Italic.ttf`, `New-Time-Bold.ttf` et `New-Time-ItalicBold.ttf`, alors vous pouvez compléter la recette avec :

```
---
fonts:
  NTime:
    regular: "fontes/New-Time-regular.ttf"
    italic: "fontes/New-Time-Italic.ttf"
    bold: "fontes/New-Time-Bold.ttf"
    italic_bold: "fontes/New-Time-ItalicBold.ttf"
```

Style personnalisé pour une fonte embarquée

Si `regular`, `italic` etc. sont des styles conventionnels qu'on peut trouver pour chaque police (et qui vont *réagir* aux marques markdown/HTML), on peut néanmoins définir un nom de style personnalisé qu'on pourra utiliser ensuite dans le texte grâce à la *stylisation en ligne* ou la programmation si vous êtes un ou une experte. Cela peut arriver, par exemple, si

vous avez acheté la licence d'une police et qu'elle contient des styles particuliers comme `demi-bold` ou `rounded`, etc.

Le style propre sera alors défini tout simplement de cette manière (comme les autres, en fait, avec un nom de style particulier et un fichier ttf/otf associé) :

```
---
fonts:
  NTime:
    monstyle: "fontes/New-Time-mes-glyphes-a-moi.ttf"
```

Alors on pourra utiliser dans le texte :

```
(( { font:'NTime', style: :monstyle } ))
```

Ce texte sera dans le style ``monstyle``, c'est-à-dire avec les glyphes de ``New-Time-mes-glyphes-a-moi.ttf``.

On pourrait imaginer par exemple que vous avez besoin de votre police, mais avec chaque lettre dans un rond. Il suffit alors de créer le fichier `New-Time-rounded.ttf` avec ces glyphes, puis de l'utiliser pour un style `rounded` :

```
---
fonts:
  NTime:
    rounded: "fontes/New-Time-mes-rounded.ttf"
```

... et vous voilà dans la possibilité d'utiliser même localement ce style avec :

Un mot aux `lettres`
entourées``.

Fonte par défaut

Noter que la fonte ci-dessus étant la toute première fonte définie dans la table `fonts`, c'est elle qui sera considérée comme la fonte par défaut et sera utilisée lorsque des polices ne seront pas définies pour des éléments du livre.

Éditeur / Maison d'édition

L'éditeur du livre (« publisher » en anglais) ou la maison d'édition se définissent dans la propriété `publisher` de la recette du livre ou de la collection.

On peut définir toutes les données utiles

```
{{TODO: Poursuivre cette fonctionnalités}}
```

Si le fichier `recipe.yaml` ou `recipe_collection.yaml` contient...

```
publisher:  
  name: "<Éditeur / Maison d'édition>"  
  adresse: "<Numéro rue\nCode Ville>"  
  contact: "<contact@chez.lui>"  
  url: "https://<url/site>"  
  logo_path: "chemin/vers/logo"
```

Le Texte du livre

Les types de paragraphes

Dans *Prawn-For-Book*, on trouve les types de paragraphe suivant ¹ :

¹ À titre de rappel, un paragraphe est un texte qui se conclut par un retour de chariot. Un paragraphe est constitué de lignes, en nombre indéfini. Une ligne est un texte qui part de la marge gauche, avec une indentation ou non, et qui s'étire jusqu'à la marge droite, si le texte est assez long.

- le **paragraphe de texte**. C'est un paragraphe normal et vous trouverez tout un tas d'indication sur son utilisation au fil de ce manuel,
- le **titre**. Cf. « Définition des titres » (p. 26),
- l'**image**. Cf. images/grand_titre (p.),
- la **table**. Cf. tables/titre_section (p.),
- le **pfb-code** (ou *code-pfb*). Cf. generalites/doubles_parentheses (p.).

Texte au format pseudo markdown

Le fichier texte du livre porte le nom « **texte.pfb.md** ».

Ce nom obligatoire permet d'utiliser l'application *Sublime Text* (<https://www.sublimetext.com>) pour rédiger son texte avec de nombreuses aides qui simplifient considérablement la tâche et permettent de se concentrer sur le contenu du livre. Notez cependant, si vous connaissez le format Markdown (ou *Multimarkdown*), que ça n'est pas du pur format Markdown et que certaines fonctionnalités ne sont pas prises en compte. Vous trouverez en annexe, page ### REF: all_markdown ###, l'intégralité des marqueurs utilisables.

Formatages du texte

Pour styliser localement le texte, c'est-à-dire le passer en italique, en gras ou en souligné, vous pouvez utiliser les marque markdown décrites ci-dessous ².

² Noter cependant que puisqu'il s'agit d'impression professionnelle, pour pouvoir utiliser de l'italique, du gras, etc., il faut que chaque style possède bien son fichier fonte (ttf ou otf) défini dans la recette. Voir « Définition des fontes (embarquées) » (p. 21).

Si le fichier « texte.pfb.md » contient...

Un **passage en italique**, un ****autre en gras****, un __troisième souligné__ et un **___***dernier avec tout***___**.

Le livre final (document PDF) contiendra :

Un *passage en italique*, un **autre en gras**, un troisième souligné et un ***dernier avec tout***.

Introduction à la stylisation en ligne

Prawn-For-Book offre une fonctionnalité très puissante, nommée « stylisation en ligne » (« inline styling » en anglais), qui permet de définir localement n'importe quel paragraphe au niveau de son aspect, de son style.

Cette *stylisation* s'inscrit juste avant le paragraphe à modifier, entre les doubles parenthèses caractéristiques de **Prawn-For-Book** et des accolades :

```
\(( \{... ici la stylisation en ligne ...} \))
```

Ici le paragraphe qui sera "touché" par la stylisation.

Pour une description détaillée de cette fonctionnalité, voir « Stylisation en ligne » (p. 49).

Si le fichier « texte.pfb.md » contient...

(({indentation: 100, color: 'FF9F00', size: 8}))

Un paragraphe qui va se retrouver indenté de 100 points-postscript, une taille de police de 8 pts et dans une couleur hexadécimale FF9F00.

Le livre final (document PDF) contiendra :

Un paragraphe qui va se retrouver indenté de 100 points-postscript, une taille de police de 8 pts et dans une couleur hexadécimale FF9F00.

Définition des titres

La format de base du texte étant le format markdown, on marque un titre dans le texte simplement grâce à des dièses suivi du titre (1 dièse => titre de niveau 1, 2 dièses => titre de niveau 2, etc.)

Le format de ces titres sont définis par défaut pour s’afficher harmonieusement avec le texte, mais on peut en définir très précisément l’aspect dans la recette, grâce à la donnée `titles` (« titres » en anglais) dans le format du livre (donnée `book_format` — « format du livre » en anglais)

```
# ./recipe.yaml ou ../recipe_collection.yaml
book_format:
  titles:
    # ... définition des titres
```

On peut commencer par y définir une *fonte-string*³ qui s’appliquera à tous les niveaux de titre, pour définir par exemple la police spéciale à utiliser (mais chaque niveau pourra ensuite définir précisément ses données).

³ Cf. *annexe/font_string* (p.)

Par exemple :

```
book_format:
  titles:
    font: "Helvetica/normal//222222"
```

Noter ci-dessus que la taille (troisième membre de la fonte-string) n’est pas définie.

Ci-dessus, par défaut, tous les titres seront en Helvetica, de style normal, et de couleur noir très légèrement éclaircie (222222).

Niveaux des titres

Dans cette section de la recette (cf. ci-dessus), on définit chaque niveau de titre grâce à la propriété `level` (« niveau » en anglais) suivi du niveau de titre, de 1 à 7 (`level1`, `level2`, `level3`...).

Astuce : si vous n’utilisez que 4 niveaux de titre, vous pouvez utiliser les titres restants (5 à 7) pour formater des titres tout à fait spéciaux.

On trouve donc:

```
book_format:
  titles:
    level1:
      # Données du titre de premier niveau
    level2:
      # Données du titre de second niveau
    level3:
```

```
# Données du titre de troisième niveau
# etc.
```

Données des niveaux de titres

Chaque niveau de titre peut définir les données/propriétés ci-dessous, à commencer par définir les données de fonte (police).

- **font** | La *fonte-string* à utiliser pour le niveau de titre en question, si elle doit être radicalement différente de la fonte par défaut des titres (cf. ci-dessus). Ou on peut se limiter à définir la taille par `//42/` mais dans ce cas, peut-être vaudrait-il mieux utiliser la propriété `size` ci-dessous.
- **size** | Pour définir la taille de police, en points.
- **style** | Pour (re)définir le style de la police. Sinon, c'est le style par défaut qui sera utilisé. Ce style, bien entendu, doit être défini dans les fontes du livre ⁴.
- **color** | Pour (re)définir la couleur du titre. Sinon, c'est la couleur par défaut qui sera utilisée.

⁴ Cf. « Définition des fontes (embarquées) » (p. 21).

On peut également indiquer l'emplacement du titre en terme de page :

- **next_page** | Si `true` (« vrai » en anglais), le titre sera placé sur une nouvelle page (paire ou impaire en fonction du flux du texte). Par défaut, seul le titre de premier niveau (`level1`) est placé sur une nouvelle page.
- **belle_page** | Si `true`, le titre sera toujours placé sur une *belle page*, c'est-à-dire une page impaire, à droite. Par défaut, seul le titre de premier niveau est placé sur une *belle page*.
- **alone** | (« seul » en anglais) Si `true` le titre sera seul sur la page (`next_page` sera automatiquement mis à `true`). Le texte qui suit le titre sera placé sur la page suivante. Dans le cas d'un titre seul, la propriété `lines_before` (« lignes avant » en anglais) permet de placer le titre dans la page (plus ou moins haut). Par défaut, seul le titre de premier niveau est placé seul sur une page.

On peut ensuite définir les lignes qu'il faut laisser avant et après le titre grâce aux propriétés :

- **lines_before** | (« lignes avant » en anglais) Nombre de lignes avant le titre.
- **lines_after** | (« lignes après » en anglais) Nombre de lignes après le titre.

Les *lignes* dont il est question ci-dessus sont des *lignes de référence* ⁵, car les textes sont toujours alignés, dans un livre bien formaté, et il en va de la même manière pour les titres.

⁵ Cf. `comportement/align_on_reference_lines` (p.).

Prawn-For-Book, c'est sa nature, adopte autant que possible un comportement intelligent par rapport à ces définitions. Par exemple, si le titre se trouve en haut de page — et que ça n'est pas un titre seul dans la page (cf. ci-dessus `alone`) — alors il n'ajoutera pas de lignes avant même si elles sont définies.

De la même manière, quand deux titres se suivent, les lignes ne s'additionnent pas (ce qui laisserait des écarts trop importants, disgracieux). C'est la valeur la plus grande qui est appliquée. Par exemple, si le titre précédent définit « 4 lignes après le titre »

(`lines_after`: 4) et que le titre suivant définit « 3 lignes avant le titre » (`lines_before`: 3), ce ne sont pas 7 lignes qui sont laissées entre les deux titres, mais 4.

On peut également définir d'autres modifications de détail du titre :

- **align** | (« alignement » en anglais) Cette propriété permet de définir l'alignement du titre. Les valeurs peuvent être `left` (« gauche » en anglais) pour un alignement à gauche, `right` (« droite » en anglais) pour un alignement à droite ou `center` (« centre » en anglais) pour un centrage dans la page.
- **left** | (« gauche » en anglais) À la place de la propriété `align` on peut définir précisément le décalage avec la marge gauche en donnant une valeur numérique (p.e. 40) ou une dimension (p.e. 52mm) à cette propriété.
- **right** | (« droite » en anglais) À la place de la propriété `align` on peut définir précisément le décalage avec la marge droite en donnant une valeur numérique (p.e. 40) ou une dimension (p.e. 52mm) à cette propriété.
- **caps** | Si `true` (« vrai » en anglais), les titres du niveau concerné seront toujours mis en capitales.
- **leading** | (« interlignage » en anglais) Permet de définir l'interlignage, quand le titre tient sur plusieurs lignes et qu'il faut le définir précisément. Cette valeur peut être négative, pour rapprocher les lignes du titre.

Titres dans les entêtes/pieds de page

Les titres peuvent être repris dans les entêtes ou pieds de page du livre. Pour savoir comment procéder, voir `header_footer/exemple_complexe` (p.).

Exclure un titre de la table des matières

Pour exclure un titre de la table des matières, le précéder ou le faire suivre par `{no-tdm}`. Par exemple :

```
### {no-tdm} Un titre hors table des matières
ou :
### Un titre hors table des matières {no-tdm}
```

Aides & Assistants

Manuel et autres aides

Tout au long de la conception de son livre — et sa production — on peut obtenir de l'aide sur ***Prawn-For-Book*** de plusieurs façon.

La façon la plus complète consiste à ouvrir ce *manuel autoproduit* qui contient l'intégralité des fonctionnalités de ***Prawn-For-Book*** expliquées de façon détaillée. C'est incontestablement **la bible de l'application**. Pour l'ouvrir, il suffit de jouer :

```
> pfb manuel
```

On peut obtenir une aide beaucoup plus succincte, rappelant les commandes de base, en jouant au choix l'une de ces commandes :

```
> pfb
> pfb aide
> pfb -h
> pfb --help
```

Aide ***Prawn-For-Book*** en ligne de commande

On peut obtenir une aide rapide sur un sujet donné, ou un mot, ou une fonctionnalité, en développant la commande `pfb aide` :

```
> pfb aide "le mot ou l'expression recherché"
```

Après avoir lancé cette commande, ***Prawn-For-Book*** affiche tous les endroits du manuel qui contiennent l'expression recherchée (par pertinence) et permet de développer le passage.

Rechercher régulière dans l'aide

On peut même faire une recherche *régulière* avec une *expression rationnelle* (si vous ne comprenez pas, cette fonctionnalités n'est peut-être pas pour vous...). L'expression rationnelle se trouvera entre guillemets et commencera et terminera avec une balance (« / »).

```
> pfb aide "/expression à rechercher>/"
```

Quelques exemples :

- Pour rechercher deux mots qui doivent se trouver dans la même phrase, et dans l'ordre donnée : `pfb aide "/mot1(.+ ?)mot2/"`.
- Pour chercher plusieurs mots : `pfb aide "/(mot1|mot2)/"`.

- Pour chercher un mot exact, mais qui peut être au pluriel : `pfb aide "/\bmots ?\b/"` (le `\b` désigne un délimiteur de mots).

Correspondances linguistiques

Cette section présente la correspondance entre tous les termes dans votre langue (français) et les termes anglais utilisés dans *Prawn-For-Book*.

centre	center
droite	right
fonte (police)	font
gauche	left
justifier	justify
largeur	width
marge bas	bot_margin
marge bas	bottom_margin
marge bas	margin_bottom
marge droite	margin_right
marge droite	right_margin
marge gauche	margin_left
marge gauche	left_margin
marge haut	margin_top
margin haut	top_margin
taille (de police)	size

Les « Snippets »

Les « snippets »⁶ (ou *complétions*) sont particulièrement utiles lorsque l'on désire gagner du temps dans la rédaction de son livre (même si, philosophiquement, "gagner du temps" est un concept *contre-artistique*...)

Ils consistent à taper quelques lettres, parfois une seule, de jouer la touche tabulation, et un texte vient remplacer automatiquement ces lettres.

Par exemple, si nous avons un personnage qui s'appelle *Rostatopoulos* et que nous ne voulons pas taper chaque fois ce prénom (avec toutes les erreurs de typo possibles...), nous créons un snippet avec la lettre « R » majuscule et chaque fois que nous tapons « R » suivi de la touche tabulation, ***Prawn-For-Book*** remplace ce « R » par « Rostatopoulos ».

⁶ Les snippets ne sont pas à confondre avec les variables textuelles qui s'écrivent textuellement dans le texte du fichier et seront ensuite remplacées par leur valeur fixe ou dynamique. Un snippet est remplacé immédiatement dans le texte par sa valeur définie.

Exclusivement dans Sublime Text

Mais cette fonctionnalité pratique ne fonctionne que dans un IDE adapté. En l'occurrence, à l'heure où nous écrivons ces lignes, elle ne fonctionne que dans l'application Sublime Text.

Installer les snippets

Pour pouvoir fonctionner, les snippets doivent être *installés* dans l'éditeur, il ne suffit pas qu'ils soient définis. Chaque fois que vous travaillez sur un nouveau livre, il faut installer les snippets définis.

On installe ces *snippets* à l'aide de la commande (dans une console ouverte au dossier du livre) :

```
> pfb install
```

Programmer un snippet

Programmer un snippet est simplissime, il suffit d'appeler la commande **pfb snippet**. Si l'on veut gagner du temps, il suffit de lui donner d'abord les lettres à taper (par exemple « R » dans notre exemple) suivi du texte de remplacement (« Rostatopoulos » dans notre exemple). Cela donnera :

```
> pfb snippet R Rostatopoulos
```

Si le texte de remplacement contient plusieurs mots, il est indispensable de les mettre entre guillemets.

```
> pfb snippet R "Rostatopoulos Alexis Triponov"
```


Messages d'erreurs et notices

On peut signaler des erreurs (messages rouges) et des messages de notices (messages bleus) au cours de la construction du livre grâce, respectivement, aux méthodes `erreur(« message> »)` (ou `error(« message> »)`) et `notice(« message> »)`.

Ces méthodes doivent être placées seules sur une ligne, entre des doubles parenthèses.

((`erreur(« message d'erreur> »)`))

Attention : ces messages ne seront jamais gravés dans le livre, ils n'apparaîtront qu'en console lorsque l'on construira le livre.

Position

Un des grands avantages de ces messages est qu'ils indiquent clairement la source de l'erreur ou de la note. Ils indiquent le numéro de page dans le livre, ainsi que le numéro de ligne dans le fichier source ou le fichier inclus. De cette manière, on retrouve très rapidement l'endroit concerné par la note ou l'erreur.

Utilisation

On peut utiliser ces méthodes par exemple pour signaler une erreur dans le livre, qu'on ne peut pas corriger au moment où on la voit. Exemple :

((`erreur(« Il manque ici le chapitre 13 »)`))

Ou simplement pour signaler une chose qu'il faut garder en tête. Par exemple :

((`notice(« Bien s'assurer que l'image qui suit soit en haut de la page. »)`))

Exemple

À cet endroit précis nous avons placé un appel à une note avec le code :

((`notice(« Une note depuis le manuel. »)`))

Vous devriez la voir si vous lancez la annexe/reconstruction_manuel (p.).

Exemple dans `texte.pfb.md`

Un paragraphe.

((`add_notice("Une simple notice pour l'exemple.")`))

Un autre paragraphe.

((`add_erreur("Une erreur signalée.")`))

Un troisième paragraphe.

Afficher grille de référence et marges

Pour régler finement l'aspect général du livre, et notamment les marges et les *lignes de référence*, on peut demander à **Prawn-For-Book** de révéler ces marges et ces *lignes de référence* par des traits de couleur.

On peut le faire soit en utilisant les *options* en ligne de commande :

```
> pfb build -- margins --grid
```

... soit, si on veut les voir affichés pendant un long moment, en définissant les valeurs de `show_margins` et `show_grid` dans le fichier recette (du livre ou de la collection) :

```
# ./recipe.yaml ou ../recipe_collection.yaml
book_format:
  page:
    show_margins: true # mettre à false quand fini
    show_grid:     true # idem
```

Quel que soit le moyen utilisé, il faut penser à retirer ces lignes avant de graver le document final à envoyer à l'imprimerie. Dans le cas contraire, ces lignes apparaîtraient dans le livre !

Exportation du texte

Pour corriger orthographiquement le texte, par exemple avec Antidote, on peut bien sûr prendre le texte du fichier `texte.pfb.md` en faisant abstraction de tout ce qui relève de la mise en forme et des codes éventuels.

Mais lorsque ces codes « extra-texte » représentent une quantité non négligeable, il est préférable de procéder à un export du seul texte du livre.

Pour se faire, il suffit de jouer la construction du livre avec l'option `-t`.

```
> cd chemin/vers/mon/livre
> pfb build -t
```

Le texte seul du livre est alors mis dans un fichier `only_text.txt` (« seulement le texte » en anglais) à la racine du livre. C'est ce fichier qu'il faut corriger.

Pagination

La *pagination* concerne la numérotation des pages et — fonctionnalité propre à **Prawn-For-Book** — la possibilité aussi de numéroter les paragraphes.

Les types de pagination

La *pagination* concerne la numérotation des pages et — fonctionnalité propre à **Prawn-For-Book** — la possibilité aussi de numéroter les paragraphes.

Par défaut, la *pagination* du livre est automatique et *intelligente*. C'est-à-dire qu'elle ne numérote que les pages qui doivent l'être :

- ne numérote pas les pages vierges (blanches),
- ne numérote pas les pages ne contenant qu'un titre,
- ne numérote pas certaines pages communes comme la table des matières, la pages des informations de fin de livre ou la page de faux titre.

Comme pour tous les comportements par défaut de **Prawn-For-Book**, il est possible de les modifier en jouant sur les paramètres de la recette ou sur les *marques en ligne* `{{TODO: Faire un lexique et mettre ce mot dedans}}` dans le texte.

Types de numérotation

Trois types de numérotation

Il existe 3 types de numérotation dans **Prawn-For-Book** : par page, par paragraphe et hybride (page et paragraphe). Ces types affectent le foliotage des pages (ce qu'on appelle la *numérotation des pages*) mais également et surtout les références aux différents endroits du livre, par exemple dans la table des matières ou les index divers.

On règle ce type dans la propriété `pagination` (ou `numeration`) de la section `book_format: page:` :

```
---
book_format:
  page:
    pagination: pages
```

Dans un livre (pour le moment, le type de numérotation doit s'appliquer à tout un livre), on peut choisir entre :

- **numérotation par page** | C'est la numérotation traditionnelle, avec le numéro de la page. C'est le type qui devra être utilisé pour un roman par exemple. Valeur à donner à `pagination: pages`

- **numérotation par paragraphe** | Dans cette numérotation, chaque paragraphe est numéroté. Elle convient à des ouvrages assez courts, lorsque l'on doit pouvoir faire référence à des paragraphes précis. Valeur à appliquer à `pagination` : `parags`.
- **numérotation hybride** | Le problème de la numérotation par paragraphe, c'est que les chiffres deviennent très élevés pour un ouvrage d'une dimension conséquente. C'est là que la *numérotation hybride* entre en jeu. Dans cette numérotation, qui fonctionne avec les pages, chaque paragraphe est numéroté, mais en reprenant à 1 à chaque *fausse page* (page gauche paire). Et l'on fait référence à un paragraphe précis en indiquant son numéro de page et son index. Par exemple « page 12 paragraphe 8 ». Vous pouvez voir cette numérotation en action dans « Numérotation des paragraphes » (p. 37). Valeur à appliquer à `pagination` : `hybrid`.

Où servent les numérotations ?

Bien sûr, quand on parle de *pagination*, on pense surtout au numéro de page qu'on trouve en entête ou en pied de page des pages du livre.

Dans *Prawn-For-Book*, cette numérotation présentera le numéro de la page pour les types `pages` et `hybrid` et présentera le numéro du premier et du dernier paragraphe pour le type `parags`.

Mais ces *numéros* servent aussi lorsqu'on fait référence à une cible placée ailleurs dans le livre. Imaginons par exemple que tous les chapitres du livre aient été référencés, grâce à des :

`<-(chapitre_<X>)`

Dans ce cas, on peut faire référence à tel ou tel chapitre avec la marque :

`->(chapitre_<X>)`

... qui présentera une référence par page, par paragraphe ou par page et paragraphe dans le type hybride.

C'est ce que nous appellerons une *référence* dans la suite.

Format des numérotations

Le format par défaut des numérotations est le suivant :

Type	Format	Exemple
pages	"page <num>"	"page 12"
parags	"§ <num parag>"	"§ 12"
hybrid	"p. <num page> § <num parag>"	"p. 2 § 5"

Formatage de la référence dans la recette

On peut modifier le format par défaut ci-dessus dans la recette dans la partie `book_format: text: references:` en définissant la propriété `format_type pagination` donc par exemple `format_hybrid` pour le format de pagination hybride, avec numéro de page et numéro de paragraphe. Dans la définition de ce format, on utilise `_page_` pour le numéro de page et `_paragraph_` pour le numéro de paragraphe.

Par exemple, si l'on est en pagination de type `hybrid` et que l'on veut que les références soient marquées « au paragraphe xxx de la page yyy », on écrira dans la recette :

```
book_format:
  text:
    references:
      format_hybrid: "au paragraphe _paragraph_ de la page _page_"
```

Ci-dessus, `_paragraph_` sera remplacé par le numéro de paragraphe de la cible et `_page_` sera remplacé par le numéro de page.

Formatage de la référence à la volée

On peut également utiliser, ponctuellement, une marque de référence différente de celle définie, lorsque cette marque ne se justifie pas.

Par exemple, si vous voulez faire référence à un chapitre, sans citer son nom, mais juste sa page, dans un texte comme « au chapitre de la page 12 », alors vous pouvez le faire de cette manière :

Rendez-vous au chapitre de la page `->(_page_|chapitre5)` pour connaître la suite.

Note : pour mémoire, le code `->(…)` fait référence à une cible définie ailleurs. Voir `references/cross_references` (p.) pour le détail.

Dans cette utilisation, on emploie `_ref_` pour faire référence au format défini (ce qui n'a pas vraiment de sens, mais on peut le faire), on emploie `_page_` pour faire référence au numéro de la page de la cible et `_paragraph_` pour faire référence au numéro de paragraphe de la cible. Notez que dans une pagination hybride, il convient d'indiquer la page et le paragraphe car l'indication au seul paragraphe serait insuffisant.

Par exemple, si on veut que la référence ressemble à « au paragraphe 3 de la page 12 », on mettra dans la première partie de la référence : `au paragraphe _paragraph_ de la page _page_`.

Aspect physique des références

Si vous n'êtes pas un expert, vous ne pouvez pas définir la fonte (police, style, taille et couleur) d'une référence dans le texte (elle doit être identique à la fonte du texte lui-même, pour la cohérence). En revanche, il est tout à fait possible de la déterminer pour la `pages_speciales/table_des_matières` (p.), les index — cf. `pages_speciales/index_page` (p.) — ou les bibliographie — `bibliographies/customisation` (p.).

Numérotation des paragraphes

Une des fonctionnalités puissante de **Prawn-For-Book** est la capacité de numérotter les paragraphes d'un texte. Cela se révèle très pratique lorsqu'on doit faire référence à des paragraphes précis, par exemple lorsque l'analyse d'un récit ou dans un ouvrage technique. Pour ce faire, il suffit d'indiquer la pagination voulue, `parags` ou `hybrid`, dans la recette du livre (ou de la collection).

Numérotation continue

La valeur `parags` attribue à chaque paragraphe du texte un numéro unique qui s'incrémente de paragraphe en paragraphe et de feuillets en feuillets. Ce format devient vite « encombrant » dans les textes qui contiennent plus d'un millier de paragraphe.

Numérotation hybride

La numérotation *hybride* (`hybrid`) convient parfaitement aux longs textes contenant des centaines de paragraphe. Elle correspond à l'indication de la page suivi de l'indication du paragraphe dans la double page, en sachant qu'à chaque nouvelle double page (en partant de la page gauche) on reprend la numérotation des paragraphes à 1. Ainsi, l'indication de pagination « 12-5 » signifie qu'il s'agit du 5^e paragraphe de la page 12.

Format de la numérotation du paragraphe

Comme tout autre élément de **Prawn-For-Book**, on peut en garder les valeurs par défaut, qui sont déjà tout à fait adaptées au livre, ou l'on peut les redéfinir. Voir « Types de numérotation » (p. 35).

SI la recette du livre contient...

```
---
book_format:
  page:
    pagination: hybrid
  text:
    references:
      hybrid_format: '§ %{paragraph} de p. %{page}'
```

ET que le texte du livre contient...

Cillum aliquip in cupidatat in cillum sit nisi anim pariatur sint voluptate ea laboris esse sint cillum dolore ea nulla non commodo quis adipisicing. Officia cillum eiusmod in incididunt consequat in voluptate veniam ut in dolore laboris consectetur ullamco proident voluptate amet esse ut anim exercitation esse non in magna consequat fugiat. Do dolore minim qui

dolore minim enim laboris exercitation ex veniam duis commodo consectetur velit sed consequat esse velit.

Ut in officia esse ad cupidatat in duis fugiat duis ut velit fugiat enim proident irure commodo occaecat reprehenderit duis duis voluptate proident tempor qui id do adipisicing dolore adipisicing irure cupidatat dolore laborum anim ea aliqua.

Eiusmod laborum consectetur est sunt laboris officia mollit quis ad mollit incididunt commodo occaecat magna officia fugiat velit cillum est culpa ut minim.

Lorem ipsum excepteur amet qui labore ut elit ea adipisicing enim dolore voluptate ut dolor enim et magna commodo deserunt reprehenderit in excepteur enim dolor amet.

Aliquip deserunt consequat irure minim tempor in laboris sint nisi nulla officia cillum sint dolore dolore do consectetur exercitation aliquip enim incididunt minim cillum quis dolore consectetur voluptate fugiat cupidatat deserunt eiusmod dolor in.

Deserunt voluptate aute consectetur eiusmod dolor in amet cupidatat sint eiusmod commodo excepteur sit ullamco occaecat commodo minim ad veniam labore.

Nulla irure sunt irure consectetur labore irure culpa deserunt occaecat ut aliqua ullamco aliquip et aute veniam ut dolore tempor.

Dolor labore consectetur REFERENCE<-(REFERENCE) ut veniam nostrud ut dolore commodo proident laboris nostrud anim quis ex amet proident pariatur do incididunt excepteur eiusmod aliquip ullamco officia consequat ex exercitation ea cupidatat ea ut consequat pariatur do dolor id.

Dolore reprehenderit sit nostrud voluptate dolore ex consectetur dolore voluptate tempor magna proident est in qui sint tempor dolor non occaecat velit (cf. — >(REFERENCE)).

Lorem ipsum id commodo tempor laboris reprehenderit dolore tempor velit elit adipisicing qui sed elit in eiusmod proident id consequat voluptate amet voluptate cupidatat aute aliquip commodo adipisicing eu in anim quis id deserunt.

Le livre final (document PDF) contiendra :

Les pages d'un livre dont les paragraphes sont numérotés ressemblent aux pages ci-dessous. Remarquez comment la numérotation recommence à la page 2 (sur la double page 2-3 donc) et comment elle se poursuit sur la page 3.

Notez également la référence qui est fait, dans le paragraphe 6 de la page 3, à une référence du paragraphe 5 de la page 2 (« § 5 de p. 2 »).

Le format de la référence (« § 5 de p. 2 ») est défini dans la recette ci-dessus, dans `book_format: text: references: hybrid_format:`.

- ¹ Lorem ipsum excepteur amet qui labore ut elit ea adipisicing enim dolore voluptate ut dolor enim et magna commodo deserunt reprehenderit in excepteur enim dolor amet.
- ² Aliquip deserunt consequat irure minim tempor in laboris sint nisi nulla officia cillum sint dolore dolore do consectetur exercitation aliquip enim incididunt minim cillum quis dolore consectetur voluptate fugiat cupidatat deserunt eiusmod dolor in.
- ³ Deserunt voluptate aute consectetur eiusmod dolor in amet cupidatat sint eiusmod commodo excepteur sit ullamco occaecat commodo minim ad veniam labore.
- ⁴ Nulla irure sunt irure consectetur labore irure culpa deserunt occaecat ut aliqua ullamco aliquip et aute veniam ut dolore tempor.
- ⁵ Dolor labore consectetur REFERENCE ut veniam nostrud ut dolore commodo proident laboris nostrud anim quis ex amet proident pariatur do incididunt excepteur eiusmod aliquip ullamco officia consequat ex exercitation ea cupidatat ea ut consequat pariatur do dolor id.

Cillum aliquip in cupidatat in cillum sit nisi anim pariatur sint ¹
voluptate ea laboris esse sint cillum dolore ea nulla non commodo quis
adipisicing. Officia cillum eiusmod in incididunt consequat in
voluptate veniam ut in dolore laboris consectetur ullamco proident
voluptate amet esse ut anim exercitation esse non in magna consequat
fugiat. Do dolore minim qui dolore minim enim laboris exercitation ex
veniam dui commodo consectetur velit sed consequat esse velit.

Ut in officia esse ad cupidatat in dui fugiat dui ut velit fugiat enim ²
proident irure commodo occaecat reprehenderit dui dui voluptate
proident tempor qui id do adipisicing dolore adipisicing irure
cupidatat dolore laborum anim ea aliqua.

Eiusmod laborum consectetur est sunt laboris officia mollit quis ad ³
mollit incididunt commodo occaecat magna officia fugiat velit cillum
est culpa ut minim.

1

Dolore reprehenderit sit nostrud voluptate dolore ex consectetur ⁶
dolore voluptate tempor magna proident est in qui sint tempor dolor
non occaecat velit (cf. REFERENCE § 5 de p. 2).

Lorem ipsum id commodo tempor laboris reprehenderit dolore tempor ⁷
velit elit adipisicing qui sed elit in eiusmod proident id consequat
voluptate amet voluptate cupidatat aute aliquip commodo adipisicing
eu in anim quis id deserunt.

3

Aspect du numéro

L'aspect du numéro (sa police, son style, sa taille, sa couleur) peut être défini très précisément dans la recette. On utilise pour cela les paramètres suivant.

La couleur est exprimée de façon hexadécimale (voir l'explication page ((### REF: couleur ###))).
(remarquez que ci-dessous nous pouvons utiliser des variables* au lieu de répéter plusieurs fois la même information. Ici, c'est la fonte par défaut que nous avons mise dans une variable)*

Note : pour voir l'aspect d'un numéro de page et/ou de paragraphe dans une référence, voir « Types de numérotation » (p. 35).

Si le fichier recipe.yaml ou recipe_collection.yaml contient...

```
---
# Recette du livre
book_format:
  page:
    num_font: "<police>/<style>/<taille>/<couleur>"
```

Suspension de la pagination

Il est nécessaire parfois de suspendre la pagination sur quelques pages. Nous l'avons vu par exemple pour pages_speciales/dedicace (p.).

Pour ce faire, il suffit d'utiliser, seul sur une ligne, les marques ((stop_pagination)) pour interrompre la pagination et la marque ((restart_pagination)) pour la reprendre (« restart » signifie « redémarrer » en anglais).

Le livre final (document PDF) contiendra :

Ci-dessous nous allons utiliser la marque ((stop_pagination)) sur une ligne pour interrompre la pagination puis la marque ((new_page)) sur une autre ligne pour passer sur la page suivante ⁷.

⁷ Noter que c'est au moment de la création de la page que **Prawn-For-Book** doit savoir qu'il ne doit pas numéroté la page. Si la marque ((new_page)) était placée avant la marque ((stop_pagination)), alors la nouvelle page créée par la première marque serait numérotée. De la même manière, la marque ((restart_pagination)) doit être placée avant la marque ((new_page)) pour que la nouvelle page créée soit numérotée.

Cette page ne doit pas être numérotée.

Page sans pagination

On peut également supprimer la pagination de la page courante avec la marque `((no_pagination))`.

Pour obtenir la page suivante, nous avons utilisé le code :

```
(( new_page ))  
(( no_pagination ))  
(( move_to_line(12) ))  
(( {size:20} ))
```

Cette page ne possède ni numéro de page ni entête.

```
(( new_page ))
```

Cette page ne possède ni numéro de page ni entête.

Numérotation des pages vierges

Les pages vierges peuvent être numérotée en modifiant la donnée `book_format:`
`page: :no_num_if_empty:` (« no num if empty » signifie « pas de numéro si la page est vide »).

Si le fichier `recipe.yaml` ou `recipe_collection.yaml` contient...

```
---
book_format:
  page:
    no_num_if_empty: false
```

Le Texte en détail

Stylisation en ligne

Nous appelons *stylisation en ligne* ou *inline styling* en anglais la possibilité de styliser un paragraphe quelconque par une ligne de code propre à **Prawn-For-Book**, c'est-à-dire entre double parenthèse. Comme nous l'avons vu dans « Introduction à la stylisation en ligne » (p. 26), ce style se met dans une table, entre accolade.

```
(( {color: "FF0000"} ))
```

D'ores et déjà, noter que cette stylisation s'applique au paragraphe suivant et *seulement* au paragraphe suivant. C'est-à-dire au texte qui suit, jusqu'à un premier retour chariot. Ce paragraphe a été mis en rouge grâce à cette *stylisation en ligne* et vous noterez que (sans que nous ayons rien fait) le paragraphe suivant a retrouvé l'aspect normal.

Stylisation en ligne par table

Comme vous pouvez le voir, la *stylisation en ligne* se fait en définissant une table, c'est-à-dire un code simple, entre accolades : { . . . }.

Experts : sachez qu'il s'agit simplement d'une table ruby, et vous pouvez passer à la suite.

À l'intérieur de ces deux accolades, on va trouver les *propriétés* avec leur *valeur*. Ci-dessus, on trouve la *propriété* `color` (« couleur » en anglais) avec la *valeur* `FF0000`

Pour définir une propriété, on a donc son nom, suivi tout de suite de deux points (ne surtout pas mettre d'espace) et sa valeur :

```
{ <propriété>: <valeur> }
```

Chaque définition de propriété est séparée par une virgule.

```
{ <propriété1>: <valeur>, <propriété2>: <valeur>,
  <propriété3>: <valeur>, etc. }
```

La *valeur*, très souvent, est un nombre (p.e. « 12 ») ou une chaîne de caractères entre guillemets droits (p.e. « "Bonjour" »). Il peut arriver que ce soit un Symbol (un type particulier du langage Ruby reconnaissable à ses deux points au début) (p.e. « :center ») ou une constante définie par **Prawn-For-Book** ou par vous (p.e. « LINE_HEIGHT »).

La *valeur* peut être aussi une dimension avec unité. Dans ce cas, elle doit obligatoirement être mise entre guillemets droits, par exemple « "2.5cm" ». Comme vous pouvez le voir, une valeur flottante (décimale) utilise le point anglais, par la virgule française.

Mais cette valeur peut être aussi une opération (p.e. « 2 * LINE_HEIGHT »), une concaténation (p.e. « "une » + « concat" ») ou toute autre expression que Ruby

comprend (p.e. « %w{un autre jour}.join("+ »)" qui produira « un+autre+jour »)).

Liste des propriétés de la stylisation en ligne

Voici la liste des propriétés qui peuvent être appliquées au paragraphe suivant (et seulement le paragraphe suivant) :

- **size** | Taille de la police du paragraphe suivant.
- **font** | Nom de la police à utiliser. Elle doit bien sûr être définie et embarquée (voir « Définition des fontes (embarquées) » (p. 21)).
- **style** | Le style à appliquer, parmi `normal`, `italic`, `bold`, `[:italic, :bold]` ou tout autre style défini explicitement pour les fontes embarquées. Voir « Définition des fontes (embarquées) » (p. 21).
- **lines_before** | (« lignes avant » en anglais) Définit combien de lignes vides on doit laisser *avant* le paragraphe.
- **lines_after** | (« lignes après » en anglais) Définit combien de lignes vides on doit laisser *après* le paragraphe.
- **indent** (ou *indentation*) | Indentation du paragraphe suivant, avec ou sans unité (p.e. '100' ou '8mm').
- **align** | Alignement du paragraphe. Peut avoir l'une des valeurs *Symbol* suivante : `:left` (« gauche » en anglais, donc alignement à gauche), `:right` (« droite » en anglais, donc alignement à droite), `:center` (« centre » en anglais, donc centré) ou `:justify` (« justifié » en anglais donc justifié — noter que par défaut, un paragraphe est justifié, dans *Prawn-For-Book*, donc cette marque ne serait utile que dans un contexte où le paragraphe ne serait plus centré, un tableau par exemple).
- **margin_left** | (« marge gauche » en anglais) définit la marge gauche.
- **margin_right** | (« marge droite » en anglais) définit la marge droite supplémentaire laissée après le texte.
- **kerning** | (« crénage » en anglais) si la valeur est à `true` (elle l'est par défaut), Prawn gèrera de façon intelligente les espaces entre les lettres pour avoir le meilleur rendu.
- **character_spacing** | (« espace entre les lettres » en anglais)

Le livre final (document PDF) contiendra :

Le code :

```
(( {font:"Reenie", size:20, indent:"8mm", color:"FF0FF0" } ))
```

Le présent paragraphe est mis en forme par de la STYLISATION EN LIGNE qui met la police à Reenie, la taille de police à 20 pt, l'indentation à 8mm et la couleur à FF0FF0.

... produira :

Le présent paragraphe est mis en forme par de la STYLISATION EN LIGNE qui met la police à Reenie, la taille de police à 20 pt, l'indentation à 8mm et la couleur à FF0FF0.

Le code :

```
(( {margin_left: "2cm", margin_right: "2cm"} ))
```

Un paragraphe qui se trouve entre deux marges resserrée de 2 cm chacune.

... produira :

Un paragraphe qui se trouve entre deux marges resserrée de 2 cm chacune.

Le code :

```
(( {margin_left: "2cm", width: PAGE_WIDTH - 4.cm } ))
```

Le même résultat (deux marges supplémentaire de 2 centimètres) peut s'obtenir avec la propriété 'width' et un calcul.

... produira :

Le même résultat (deux marges supplémentaire de 2 centimètres) peut s'obtenir avec la propriété 'width' et un calcul.

Le code :

```
(( { character_spacing: 4, kerning: true, align: :left } ))
```

Un texte avec les lettres espacées.

(le *kerning* à `true` indique de gérer les espaces entres les lettres pour avoir le meilleur rendu)

... produira :

Un texte avec les lettres espacées.

(le *kerning* à *true* indique de gérer les espaces entres les lettres pour avoir le meilleur rendu)

Le code :

Le paragraphe avant pour voir les lignes vides après.

```
(( { lines_before: 4, lines_after: 3 } ))
```

Un texte avec un `lines_before` de 4 (donc 4 lignes vides avant) et un `lines_after` de 3 (donc avec 3 lignes vides après).

Le paragraphe après pour voir les lignes vides avant.

... produira :

Le paragraphe avant pour voir les lignes vides après.

Un texte avec un `lines_before` de 4 (donc 4 lignes vides avant) et un `lines_after` de 3 (donc avec 3 lignes vides après).

Le paragraphe après pour voir les lignes vides avant.

Indentation des paragraphes

La propriété « `book_format: text: indent:` » permet de déterminer l'indentation des paragraphes. On la définit avec une valeur numérique en points-ps ou dans toute autre valeur acceptée, comme les millimètres ('20mm') ou les pixels ('20px').

Comme pour de nombreux éléments de publication, *Prawn-For-Book* se comporte de façon intelligente avec les indentations. C'est-à-dire qu'il n'en ajoute, normalement, que là où c'est nécessaire. Il n'en met pas, par exemple, après un titre ou un paragraphe vide.

Mais comme de nombreux éléments de publication, *Prawn-For-Book* sait s'adapter aux besoins de permet de tout régler finement. Il est ainsi possible de définir explicitement une indentation qu'on veut ponctuellement différente, ou de la supprimer là où elle devrait se mettre. Étudiez les exemples suivants.

Indentation en stylisation en ligne

On peut ponctuellement introduire une indentation d'un paragraphe en utilisant la propriété `indent` ou `indentation`.

Par exemple, ce paragraphe a été précédé du code `(({indent: '2cm'}))` et se retrouve donc indenté de 2 centimètres.

Ce paragraphe, au contraire, a été précédé du code `(({indentation: 8}))` et se retrouve donc indenté de 8 points-postscript.

Ce dernier paragraphe ne subit aucune indentation, parce que pour rappel, la stylisation en ligne (cf. page) ne s'applique qu'au paragraphe suivant.

Pas d'indentation négative

Noter que pour le moment il est impossible d'utiliser une indentation négative (à cause des limitations actuelles de *Prawn*).

Forcer l'indentation du paragraphe

Il peut arriver, parfois, que l'indentation d'un paragraphe soit supprimée, sans raison apparente. Cela tient aux calculs parfois très compliqués que doit effectuer *Prawn-For-Book*. Le cas échéant, il suffit d'ajouter la marque `(({indentation:true}))` juste avant le paragraphe en question pour forcer son indentation.

Note sur l'indentation pour les experts

Le cas échéant, sachez que l'indentation des paragraphes dans *Prawn-For-Book* n'utilise pas, en vérité, la propriété `:indent_paragraphs`. Tout simplement parce cette propriété, à l'heure où l'on écrit ces lignes, n'est pas utilisable pour calculer la hauteur d'un bloc. On serait donc contraint de la supprimer, obtenant donc des résultats faux.

Pour palier ce problème, on ajoute en réalité des espaces insécables vides avant le texte, pour atteindre peu ou prou la longueur désirée.

Si le fichier `recipe.yaml` ou `recipe_collection.yaml` contient...

```
book_format:
  text:
    indent: '2cm'
```

Si le fichier « `texte.pfb.md` » contient...

Un texte normal donc sans indentation puisque nous n'en utilisons pas dans ce manuel.

(({indentation: '40mm'}))

Ce paragraphe, ponctuellement, possède une indentation de 40 mm définie par la ligne de code au-dessus de lui.

Et ce troisième paragraphe utilise à nouveau l'indentation par défaut (donc pas d'indentation).

 Et normal.

(({no_indentation: true}))

Ce dernier paragraphe, précédé d'un pfb-code supprimant son indentation, est imprimé contre la marge gauche (mais bon... c'est un test gratuit puisqu'il n'y a pas d'indentation par défaut...).

Le livre final (document PDF) contiendra :

Un texte normal donc sans indentation puisque nous n'en utilisons pas dans ce manuel.

Ce paragraphe, ponctuellement, possède une indentation de 40 mm définie par la ligne de code au-dessus de lui.

Et ce troisième paragraphe utilise à nouveau l'indentation par défaut (donc pas d'indentation).

Et normal.

Ce dernier paragraphe, précédé d'un pfb-code supprimant son indentation, est imprimé contre la marge gauche (mais bon... c'est un test gratuit puisqu'il n'y a pas d'indentation par défaut...).

Gestion des tirets conditionnels

On peut insérer très simplement des tirets conditionnels⁸ à l'aide de la marque dans le texte, à l'endroit où doit se trouver un tiret conditionnel.

⁸ Pour rappel, un tiret conditionnel permet d'indiquer explicitement où doit se faire une césure (une coupure de mot en fin de ligne à droite) si cette césure est nécessaire et seulement si cette césure est nécessaire, ce qui fait son intérêt. Lorsque le mot est long, on peut placer plusieurs tirets conditionnels qui laisseront **Prawn-For-Book** choisir le meilleur en fonction du contexte.

Si le fichier « `texte.pfb.md` » contient...

Dans ce texte, vers la fin de la ligne on a anti{-}cons{-}ti{-}
 }tu{-}tion{-}nel{-}lement qui peut s'adapter grâce aux tirets conditionnels.

Sans les tirets conditionnels :

Dans ce texte, vers la fin de la ligne on a
 anticonstitutionnellement qui NE peut PAS s'adapter.

Le livre final (document PDF) contiendra :

Dans ce texte, vers la fin de la ligne on a anticonstitutionnel-
 lement qui peut s'adapter grâce aux tirets conditionnels.

Sans les tirets conditionnels :

Dans ce texte, vers la fin de la ligne on a
 anticonstitutionnellement qui NE peut PAS s'adapter.

Placement sur une ligne quelconque

Grâce aux *codes en ligne* de **Prawn-For-Book**, on peut se placer sur une ligne quelconque de la page à l'aide de : `move_to_line(x>)`.

Si le fichier « `texte.pfb.md` » contient...

Le code ci-dessous permet de placer le paragraphe sur la dernière ligne de la page.

```
(( move_to_line(-1) ))
```

Ce paragraphe est placé sur la dernière ligne.

Le livre final (document PDF) contiendra :

Le code ci-dessous permet de placer le paragraphe sur la dernière ligne de la page.

Ce paragraphe est placé sur la dernière ligne.

Les puces

On peut utiliser de nombreuses puces et les régler parfaitement à sa convenance en modifiant la recette du livre ou non.

Si le fichier « texte.pfb.md » contient...

* Ceci est la puce de base qui permet un affichage neutre des items de liste.

Le livre final (document PDF) contiendra :

– Ceci est la puce de base qui permet un affichage neutre des items de liste.

Puce losange

SI la recette du livre contient...

```
---
book_format:
  text:
    puce:
      text: :losange
      size: 18
      left: 20mm
      vadjust: 2mm
```

ET que le texte du livre contient...

* Ceci est une puce :losange de 18 points, remontée de 2 points, avec les autres paramètres laissés intacts,
 * Le second item identique,
 * Le troisième.

ALORS le livre contiendra...

- ◇ Ceci est une puce : `losange` de 18 points, remontée de 2 points, avec les autres paramètres laissés intacts,
- ◇ Le second item identique,
- ◇ Le troisième.

1

(Noter dans la recette l'ajustement vertical de la puce grâce à la propriété `vadjust` mise à 2)

Puce losange noir

SI la recette du livre contient...

```
---
book_format:
  text:
    puce:
      text: :black_losange
      size: 14
      vadjust: 2mm
```

ET que le texte du livre contient...

* Ceci est une puce :black_losange de 14 points, remontée de 2 points, avec les autres paramètres laissés intacts,
 * Le second item identique,
 * Le troisième.

ALORS le livre contiendra...

- ◆ Ceci est une puce :black_losange de 14 points, remontée de 2 points, avec les autres paramètres laissés intacts,
- ◆ Le second item identique,
- ◆ Le troisième.

(Noter dans la recette l'ajustement vertical de la puce grâce à la propriété vadjust mise à 2)

Puce carrée

SI la recette du livre contient...

```
---
book_format:
  text:
    puce:
      text: :square
      size: 20
      vadjust: 2mm
```

ET que le texte du livre contient...

* Ceci est une puce :square de 20 points, remontée de 2 points, avec les autres paramètres laissés intacts,
* Le second item identique,
* Le troisième.

ALORS le livre contiendra...

- ❑ Ceci est une puce :square de 20 points, remontée de 2 points, avec les autres paramètres laissés intacts,
- ❑ Le second item identique,
- ❑ Le troisième.

1

(Noter dans la recette l'ajustement vertical de la puce grâce à la propriété vadjust mise à 2)

Puce carrée noire

SI la recette du livre contient...

```
---
book_format:
  text:
    puce:
      text: :black_square
      size: 20
      vadjust: 2mm
```

ET que le texte du livre contient...

* Ceci est une puce :black_square de 20 points, remontée de 2 points, avec les autres paramètres laissés intacts,
 * Le second item identique,
 * Le troisième.

ALORS le livre contiendra...

- Ceci est une puce :black_square de 20 points, remontée de 2 points, avec les autres paramètres laissés intacts,
- Le second item identique,
- Le troisième.

(Noter dans la recette l'ajustement vertical de la puce grâce à la propriété vadjust mise à 2)

Puce ronde

SI la recette du livre contient...

```
---
book_format:
  text:
    puce:
      text: :bullet
      size: 20
      vadjust: 4mm
```

ET que le texte du livre contient...

* Ceci est une puce `:bullet` de 20 points, remontée de 4 points, avec les autres paramètres laissés intacts,
* Le second item identique,
* Le troisième.

ALORS le livre contiendra...

- Ceci est une puce `:bullet` de 20 points, remontée de 4 points, avec les autres paramètres laissés intacts,
- Le second item identique,
- Le troisième.

(Noter dans la recette l'ajustement vertical de la puce grâce à la propriété `vadjust` mise à 4)

Puce ronde noire

SI la recette du livre contient...

```
---
book_format:
  text:
    puce:
      text: :black_bullet
      size: 30
      left: 6.5mm
      vadjust: 10mm
```

ET que le texte du livre contient...

* Ceci est une puce :black_bullet de 30 points, remontée de 10 points, avec les autres paramètres laissés intacts,
 * Le second item identique,
 * Le troisième.

ALORS le livre contiendra...

- Ceci est une puce :black_bullet de 30 points, remontée de 10 points, avec les autres paramètres laissés intacts,
- Le second item identique,
- Le troisième.

(Noter dans la recette l'ajustement vertical de la puce grâce à la propriété vadjust mise à 10)

Puce doigt tendu

SI la recette du livre contient...

```
---
book_format:
  text:
    puce:
      text: :finger
      size: 20
      left: 6.5mm
      vadjust: 1mm
```

ET que le texte du livre contient...

- * Ceci est une puce :finger de 20 points, remontée de 1 point, avec les autres paramètres laissés intacts,
- * Le second item identique,
- * Le troisième.

ALORS le livre contiendra...

- ☞ Ceci est une puce :finger de 20 points, remontée de 1 point, avec les autres paramètres laissés intacts,
- ☞ Le second item identique,
- ☞ Le troisième.

(Noter dans la recette l'ajustement vertical de la puce grâce à la propriété `vadjust` mise à 1)

Puce doigt tendu noir

SI la recette du livre contient...

```
---
book_format:
  text:
    puce:
      text: :black_finger
      size: 20
      left: 7mm
      vadjust: 3mm
```

ET que le texte du livre contient...

- * Ceci est une puce :black_finger de 20 points, remontée de 3 points, avec les autres paramètres laissés intacts,
- * Le second item identique,
- * Le troisième.

ALORS le livre contiendra...

- ☛ Ceci est une puce :black_finger de 20 points, remontée de 3 points, avec les autres paramètres laissés intacts,
- ☛ Le second item identique,
- ☛ Le troisième.

(Noter dans la recette l'ajustement vertical de la puce grâce à la propriété vadjust mise à 3)

Puce gros tiret

SI la recette du livre contient...

```
---
book_format:
  text:
    puce:
      text: :hyphen
      size: 100
      left: 20mm
      vadjust: 42mm
```

ET que le texte du livre contient...

- * Ceci est une puce :hyphen de 100 points, remontée de 42 points, avec les autres paramètres laissés intacts,
- * Le second item identique,
- * Le troisième.

ALORS le livre contiendra...

████████ Ceci est une puce :hyphen de 100 points, remontée de 42 points, avec les autres paramètres laissés intacts,

████████ Le second item identique,

████████ Le troisième.

(Noter dans la recette l'ajustement vertical de la puce grâce à la propriété vadjust mise à 42)

Puce personnalisée

Enfin, on peut utiliser des puces personnalisées partant d'une image que vous fournissez. Il suffit de donner son chemin d'accès soit relatif, par rapport au dossier du livre ou de la collection (recommandé), soit absolu. Dans l'exemple, la puce se trouve dans le dossier 'images' du livre qui, comme son nom l'indique, contient toutes les images que nous utilisons dans le manuel.

On peut aussi définir la propriété `:height` pour la hauteur de l'image. Si `:width` est défini (et non proportionnel), l'image sera déformée, sinon, la largeur sera proportionnellement adaptée à la hauteur.




SI la recette du livre contient...

```
---
book_format:
  text:
    puce:
      text: "images/custom_bullet.png"
      size: 14
      left: 8mm
      vadjust: 2mm
```

ET que le texte du livre contient...

- * Ceci est une puce "images/custom_bullet.png" de 14 points, remontée de 2 points, avec les autres paramètres laissés intacts,
- * Le second item identique,
- * Le troisième.

ALORS le livre contiendra...

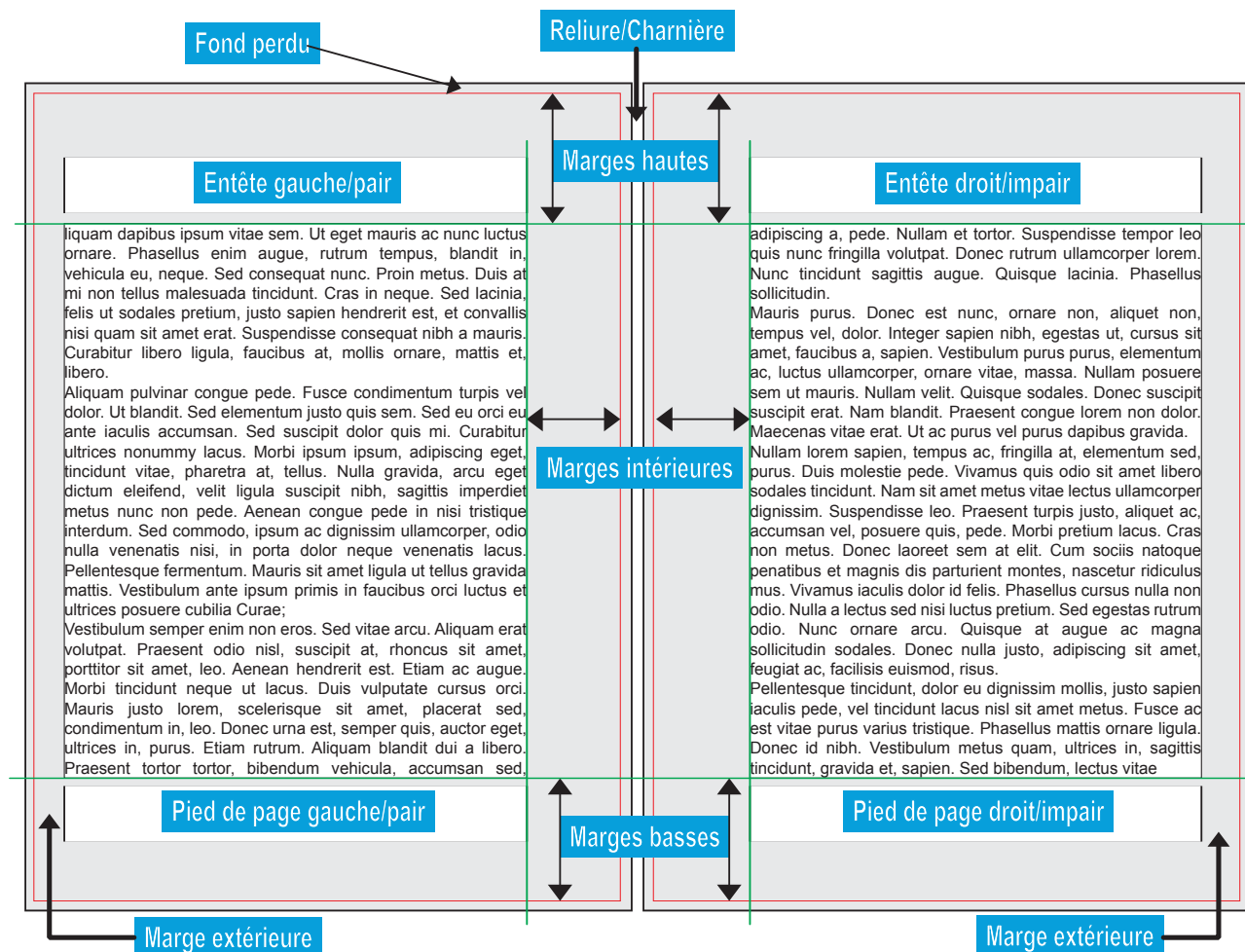
-  Ceci est une puce "images/custom_bullet.png" de 14 points, remontée de 2 points, avec les autres paramètres laissés intacts,
-  Le second item identique,
-  Le troisième.

1

(Noter dans la recette l'ajustement vertical de la puce grâce à la propriété `vaadjust` mise à 2)

Format du livre et des pages

Définition des marges



Des marges par défaut sont proposées, mais vous pouvez tout à fait définir celles que vous voulez très précisément, dans la recette du livre ou de la collection.

La seule chose à comprendre, par rapport aux documents dont vous avez l'habitude, c'est qu'ici les pages sont paires ou impaires, en vis-à-vis, et définissent donc :

- une marge haute et une marge basse (traditionnelle),
- une marge *intérieure*, qui comme son nom l'indique est à l'intérieur du livre, près de la reliure, de la *charnière*, du *dos du livre* (souvent confondu avec la *tranche du livre*),
- une marge *extérieure*, qui comme son nom l'indique est tournée vers l'extérieur du livre, vers la tranche (la vraie cette fois).

Ces marges sont donc définies par les propriétés `top` (« haut » en anglais) `bot` (pour « bottom », « bas » en anglais), `ext` (pour « extérieur ») et `int` (pour « intérieur »).

NB : Quels que soient les réglages, il y aura toujours un *fond perdu* (« bleeding » en anglais) de 10 pps (points-postscript) autour de la page. C'est la « marge » que s'accorde l'imprimeur pour découper le livre.

Traditionnellement, la marge intérieure est plus large que la marge extérieure, car une bonne partie de cette marge est prise dans la reliure.

De la même manière, la marge basse est plus large que la marge haute car elle contient le numéro de page. Il peut cependant arriver que la marge haute contienne un entête.

Pour régler parfaitement les marges, vous pouvez soit ajouter l'option `-margins` à la commande `pfb build` qui construit le livre, soit mettre le `show_margins` de la recette à `true`, comme nous l'avons fait ci-dessous.

Dans l'exemple ci-dessous nous avons volontaire *pousser* les valeurs pour rendre bien visibles les changements.

SI la recette du livre contient...

```
book_format:
  page:
    margins:
      top: 40mm
      bot: 20mm
      ext: 2mm
      int: 35mm
    show_margins: true
```

ET que le texte du livre contient...

Pour cette page, où les marges sont visibles, on illustre des marges de page à 40 mm en haut, 20 mm en bas, 35 mm à l'intérieur et 2 mm à l'extérieur.

Vous remarquez donc une immense marge en haut, une grande marge en bas, une marge externe toute petite (*ce qui ne serait pas du tout bon pour une impression de livre*) et une marge intérieure moyenne.

Remarquez aussi que le texte est automatiquement justifié, il s'aligne parfaitement sur le bord de ces marges gauche et droite, ce qui donne un rendu impeccable.

ALORS le livre contiendra...

Les marges sont visibles grâce au `show_margins` mis à `true` dans la recette. On aurait pu aussi jouer la commande avec l'option `-margins`.

Pour cette page, où les marges sont visibles, on illustre des marges de page à 40 mm en haut, 20 mm en bas, 35 mm à l'intérieur et 2 mm à l'extérieur.

Vous remarquez donc une immense marge en haut, une grande marge en bas, une marge externe toute petite (*ce qui ne serait pas du tout bon pour une impression de livre*) et une marge intérieure moyenne.

1

Remarquez aussi que le texte est automatiquement justifié, il s'aligne parfaitement sur le bord de ces marges gauche et droite, ce qui donne un rendu impeccable.

2

Section en multi-colonnes

On peut provisoirement passer en double colonnes grâce à la marque :

```
(( colonnes(2) ))
```

Pour arrêter d'être en double colonnes, il suffit d'utiliser :

```
(( colonnes(1) ))
```

Vous l'aurez déjà compris, grâce à cette marque, on peut avoir autant de colonnes que l'on désire.

Définition plus précise des colonnes

On peut définir la gouttière (espace entre chaque colonne) grâce à la propriété `gutter` à mettre en deuxième paramètre :

```
(( colonnes(2, gutter: 40) ))
```

On peut définir aussi sur quelle largeur les colonnes devront tenir, par exemple la moitié de la page :

```
(( colonnes(2, width: PAGE_WIDTH/2) ))9
```

⁹ Vous remarquez ci-dessus l'utilisation d'une constante (cf. annexe/constantes (p.)).

Précaution pour les colonnes

Attention à toujours terminer par `((colonnes(1)))`, surtout si c'est la dernière page, dans le cas contraire les pages multi-colonnes ne seraient pas gravées.

Le livre final (document PDF) contiendra :

Un premier paragraphe qui commence en simple colonne. Juste sous ce paragraphe, on a inscrit le code (invisible ici) : `((colonnes(3)))` qui permet de passer la suite en triple colonnes.

Début du texte. In mollit veniam est ut officia sit mollit est dolor consequat anim veniam est ut officia sit mollit est dolor consequat cillum. In mollit anim mollit est dolor consequat cillum. In mollit anim veniam est ut officia sit mollit est dolor consequat cillum. In mollit anim veniam est ut officia sit mollit est dolor consequat cillum. In mollit anim veniam est ut officia sit mollit est dolor consequat cillum. In mollit anim veniam est ut officia sit mollit est dolor consequat cillum. In mollit anim veniam est ut officia sit mollit est dolor consequat

cillum. In mollit anim
veniam est ut officia sit
mollit est dolor consequat
cillum. In mollit anim
veniam est ut officia sit
mollit est dolor consequat
cillum. In mollit anim
veniam est ut officia sit
mollit est dolor consequat
cillum. In mollit anim
veniam est ut officia sit
mollit est dolor consequat

cillum. In mollit anim
veniam est ut officia sit
mollit est dolor consequat
cillum. In mollit anim
veniam est ut officia sit
mollit est dolor consequat
cillum. In mollit anim
veniam est ut officia sit
mollit est dolor consequat
cillum. In mollit anim
veniam est ut officia sit
mollit est dolor consequat

cillum. In mollit anim
veniam est ut officia sit
mollit est dolor consequat
cillum. In mollit anim
veniam est ut officia sit
mollit est dolor consequat
cillum. In mollit anim
veniam est ut officia sit
mollit est dolor consequat
cillum. Fin du texte.

On revient ensuite à un texte sur une colonne avec la marque `((colonnes(1)))`. Et c'est la fin de l'usage des colonnes multiples, on revient sur une page normale.

La double colonne suivante est obtenue quant à elle grâce au code : `((colonnes(2, width:PAGE_WIDTH/1.5, gutter:50)))` qui est placé juste sous cette ligne.

Début du texte. In mollit
anim veniam est ut officia
sit mollit est dolor
consequat cillum. In
mollit anim veniam est ut
officia sit mollit est dolor
consequat cillum. In
mollit anim veniam est ut
officia sit mollit est dolor
consequat cillum. In
mollit anim veniam est ut
officia sit mollit est dolor
consequat cillum. In
mollit anim veniam est ut

officia sit mollit est dolor
consequat cillum. In
mollit anim veniam est ut
officia sit mollit est dolor
consequat cillum. In
mollit anim veniam est ut
officia sit mollit est dolor
consequat cillum. In
mollit anim veniam est ut
officia sit mollit est dolor
consequat cillum. . Fin du
texte.

On revient à nouveau à un texte sur une colonne avec la marque `((colonnes(1)))`. Par défaut, **Prawn-For-Book** laisse une ligne vide au-dessus et une ligne vide au-dessous d'un texte en multi-colonnes. On peut contrecarrer ce comportement à l'aide des propriétés `lines_before` et `lines_after`. À `false` ou 0, aucune ligne ne sera ajoutée.

On peut même mettre `line_after` à -1 lorsqu'il arrive, parfois, qu'une ligne supplémentaire soit ajoutée par erreur (les calculs de Prawn sont parfois impénétrables). Avant les doubles colonnes suivantes, nous avons écrit le code : `((columns(2, lines_before:0, space_after: - LINE_HEIGHT)))` (nous avons volontairement utilisé la traduction anglaise « columns »).

Début du texte en double colonnes. Un dessous avec le texte après. Tous ces textes devraient être collés. Fin du texte en double colonnes.

ni au-dessus avec le texte avant ni au-Paragraphe sous le texte en double colonnes collées. Ci-dessus, nous avons dû jouer sur `space_after`, avec une valeur négative, pour arriver à nos fins car `lines_after` restait inefficace. Au-dessus, on peut aussi jouer avec `space_before` si on veut définir l'espace avant. Notez que le texte est quand même remis sur des lignes de référence à chaque fois.

Ligne en trop dans les multi-colonnes

Parfois il peut arriver que **Prawn-For-Book** compte une ligne de trop dans les colonnes, ce qui produit cet alignement pas très heureux :

Premier	Deuxième en regard
Deuxième	Troisième en regard
Troisième	
Premier en regard	

Pour palier cet écueil, on met la propriété `no_extra_line_height` à `true` dans la définition des colonnes. On obtient alors :

Premier	Premier en regard
Deuxième	Deuxième en regard
Troisième	Troisième en regard

Le Mode Expert

Description

Le *mode Expert* permet d'élargir de façon exponentielle les possibilités de **Prawn-For-Book** afin de produire les contenus les plus variés et les plus originaux.

Il demande une compétence particulière dans le langage Ruby ainsi qu'une bonne connaissance de la gem Prawn qui permet de produire les livres avec **Prawn-For-Book**.

Indication du contexte d'erreur

À titre préventif dans les méthodes personnalisées, *helpers* et autres modules, on peut indiquer le contexte qui devra être affiché en cas d'erreur.

Cela se fait en utilisant le code `PFBError.context = "Le contexte"`.

```
# ruby
def monHelper(pdf, book)
  12.times do |i|
    # Indiquer le contexte
    PFBError.context = <<~EOC
      Dans la boucle de calcul et d'écriture du
      chiffre, avec i = \\#{i}
    EOC
    ecrire_ce_chiffre(i)
  end
  # Penser à "défaire" le contexte
  PFBError.context = nil
end
```

Référence à un paragraphe

Il peut arriver qu'on ait besoin de faire référence à un paragraphe spécial au cours du formatage du texte.

Pour ce faire, en *mode expert*, on peut utiliser la méthode ruby `#reference` des paragraphes.

Par exemple :

```
def methode_formatage(str, context)
  paragraph = context[:paragraph]
end
```

Si on veut juste le numéro de la page ou du paragraphe (sans préfixe), on ajouter `false`

(« faux » en anglais)

Par exemple :

```
def methode_formatage(str, context)
  paragraph = context[:paragraph]
  "{str} est situé à la page {paragraph.reference\(false)}."
end
```

Injection

book.inject(pdf, string, idx, self) est vraiment la formule magique pour ajouter du contenu au livre. L'avantage principale de cette méthode est d'analyser précisément le type de contenu — représenté ici par `string` — et de le traiter conformément à son type. Par exemple :

- si `string` est " ![images/mon.svg]", alors ce sera une image qui sera traitée,
- si `string` est "### Mon beau titre" alors c'est un titre qui sera inséré,
- si `string` est "((new_page))" alors on passera à la nouvelle page ¹.

¹ Bien sûr, ici, dans le programme, on pourrait utiliser `pdf.start_new_page`, mais l'idée derrière l'utilisation de `book.inject(...)` est de pouvoir utiliser le même code que dans le livre. Inutile d'apprendre une nouvelle langue ou de fouiller dans le code du programme pour savoir comment exécuter telle ou telle action.

idx correspond à l'index du paragraphe dans la source injectée, il n'a de valeur que pour le débogage. Dans le programme, il correspond par exemple au numéro de ligne dans le fichier. On pourra l'utiliser comme l'on veut.

self correspond dans le programme à l'instance du fichier de texte (`Prawn4book::InputTextFile`). On peut le définir comme tel si le code injecté vient d'un fichier (même si, dans ce cas, il vaudrait mieux utiliser tout simplement la `string` : `((include mon/fichier.md))`). Si elle n'est pas fournie, elle sera égale à `"user_metho"`.

{{TODO: Développer encore}}

Évaluation du code ruby

Tous les codes qui se trouveront entre « `#{...}` » (ou entre « `#{{{...}}}` » lorsque le code contient des accolades) seront évalués en tant que code ruby, dans le cadre du livre (c'est-à-dire qu'ils pourront faire appel à des méthodes personnalisées).

Typiquement, on peut par exemple obtenir la date courante ou le numéro de version du livre

pour l'insérer dans les premières pages à titre de repère, comme vous pouvez le voir dans l'exemple ci-dessous.

Code ruby sans retour chariot

Pour le moment, on ne peut pas utiliser de retours chariot dans le code ruby à évaluer. Les remplacer par des points virgules pour utiliser plusieurs lignes. Par exemple :

Ce code `#{n = 3; 12.times do |i|; n += i; end}`

Utilisation unique des triples accolades

Noter que contrairement à l'utilisation simple `#{...}` qu'on peut trouver plusieurs fois par paragraphe, l'utilisation des triples accolades `#{{{...}}}` ne doit se faire impérativement qu'une seule fois dans un même paragraphe.

Retour du code

Il faut garder en tête que le retour du code produit s'inscrit dans la page. Si vous voulez exécuter une opération qui ne doit pas produire de texte à inscrire, vous avez la solution...

... soit d'ajouter `nil` ou `" "` à la suite du code (après `<< ; >>` évidemment) :

ce code `#{2 + 2; nil}` n'écrira rien.

... soit de commencer le code par le signe moins :

ce code `#{- 2 + 2}` n'écrira rien.

Évaluation au second tour

Certaines opérations ou certains calculs ne peuvent s'opérer qu'au second tour de l'application¹⁰ — typiquement, le nombre de pages du livre —. On utilise alors la tournure suivante pour réaliser ces opérations.

`#{{{ "#{operation>}" if Prawn4book.second_turn?}}}`¹¹

Dans le code ci-dessus, le contenu des guillemets ne sera évalué qu'au second tour de l'application. Mais attention, cela peut occasionner un changement des numéros de page si le texte ajouté au second tour est conséquent. Il est donc plus prudent de mettre au premier tour un texte d'environ la longueur du résultat attendu pour ne pas fausser le suivi. Pour le numéro des pages, que nous estimons au départ à plusieurs centaines mais moins d'un millier nous utilisons :

`#{{{ Prawn4book.first_turn? ? « XXX » :
« #{book.pages.count} » }}}}`

... qui signifie qu'au premier tour, *Prawn-For-Book* marquera simplement « XXX » et aux suivants il inscrira le nombre de pages.

¹⁰ *C'est le cas par exemple de l'impression du nombre de pages de ce manuel dans l'avant-propos, c'est-à-dire alors*

que le livre est à peine esquissé.

¹¹ Noter ici l'utilisation des trois accolades obligatoires lorsque le code lui-même a recours aux accolades.

Si le fichier « `texte.pfb.md` » contient...

J'utilise actuellement la version `#{RUBY_VERSION}` de ruby. Une opération simple permet de savoir que `2 + 2` est égal à `#{2+2}` et que le jour courant (au moment de l'impression de ce livre) est le `#{Time.now.strftime('%d %m %Y')}`. Je suis juste un calcul sans écriture`#{- 2 + 2}`.

Le livre final (document PDF) contiendra :

J'utilise actuellement la version 2.7.3 de ruby. Une opération simple permet de savoir que `2 + 2` est égal à `4` et que le jour courant (au moment de l'impression de ce livre) est le 14 01 2024. Je suis juste un calcul sans écriture.

Méthodes d'helpers en mode expert

Une fonctionnalité très pratique concerne ce qu'on appelle en ruby les *helpers*, c'est-à-dire (comme vous devez le savoir en tant qu'expert) des méthodes qui permettent de mettre en forme des textes récurrents.

Imaginons qu'on veuille par exemple un format d'horloge (donnant l'heure) très particulier dans le texte. Plutôt que de répéter ce formatage pour chaque horloge à écrire, on va créer un *helper* qui va faire l'opération pour nous.

Cet *helper* devra être une méthode d'instance de `PdfBook::AnyParagraph` (ou `PdfBook::NTextParagraph` s'il ne concerne que les paragraphes de texte) du module `Prawn4book` par exemple dans le fichier `prawn4book.rb` ou `formater.rb`.

Anti-collision pour les noms de méthode d'helper

Pour le moment, les méthodes d'helper se situant à un haut niveau des instances, il peut y avoir *collision de nom* (quand le nom de l'helper est déjà utilisé par l'instance) ce qui détraquerait complètement la construction du livre.

L'astuce pour être sûr d'éviter cette erreur est simple : il suffit d'utiliser une première fois ce helper dans votre texte, comme s'il existait, mais sans implémenter l'helper (ou en le mettant dans un commentaire, comme s'il n'existait pas). Si le programme provoque une erreur (de méthode inconnue ou il se bloque), alors vous savez que votre méthode est unique, qu'il n'y a pas collision.

Vous pouvez alors décommenter votre helper pour l'utiliser.

Par exemple dans `./formater.rb`

```

module Prawn4book
  class PdfBook::AnyParagraph
    def horloge hhmmss
      h, m, s = hhmmss.split(':').map { |n| n.to_i }
      [h + __s(h), m + __s(m), s + __s(s)].join(' ')
    end
    #
    def __s val
      val.to_i > 1 ? "s" : ""
    end
  end
end
end

```

Si le fichier « texte.pfb.md » contient...

Un texte avec une horloge à #{horloge('1:12:02')} et une autre à #{horloge('0:1:1')} pour terminer à #{horloge('3:03:15')}.

Le livre final (document PDF) contiendra :

Un texte avec une horloge à 1 12s 2s et une autre à 0 1 1 pour terminer à 3s 3s 15s.

Parsing du paragraphe

Un texte de paragraphe est souvent constitué de variables, de code, qui dépendent du contexte ou sont définis pour un livre. En règle générale, ces variables sont estimées au cours de la fabrication du livre.

Mais parfois, il est nécessaire de forcer cette interprétation lorsque beaucoup de texte est produit de façon informatique (par programmation), comme par exemple un dictionnaire qui serait produit à partir d'une base de données de mots.

Dans ce cas, il faut explicitement appeler la méthode :

```
Prawn4book::PdfBook::AnyParagraph.__parse
```

... en lui fournissant les bons arguments :

```
str_corrige = \
  Prawn4book::PdfBook::AnyParagraph.__parse(<string>, <context>)
```

où :

<string> est la chaîne produite [String].

<context> est une table [Hash] définissant :pdf et :paragraph ¹²

¹² Rappel : pour obtenir :pdf et :paragraph, se souvenir que plusieurs méthodes de parsing et de formatage personnalisées peuvent utiliser leur nombre de paramètres pour déterminer les informations qui seront transmises.

Bibliographies en mode expert

TODO: Décrire comment faire une méthode de formatage propre dans `Prawn4book::Bibliography` (méthode d'instance) pour l'utiliser dans la liste des sources, pour une propriété particulières. Si, par exemple, la donnée `format` de la bibliographie (dans la recette), définit :

```
%{title|mon_transformeur}
```

... alors il faut définir la méthode :

```
Prawn4book::Bibliography#mon_tranformeur
```

... qui reçoit en argument la valeur de `:title` de l'item.

TODO: Montrer qu'on peut par exemple définir une font et/ou une couleur propre comme pour la bibliographie `costume`

Mode Multi-colonnes

En tant qu'expert, vous pouvez utiliser le mode multi-colonnes (affichage du texte sur plusieurs colonnes) à l'intérieur des formateurs.

Bien entendu, vous pouvez, si vous êtes parfaitement à l'aise avec ça, utiliser les `column_box` de **Prawn**. Mais ***Prawn-For-Book*** propose là aussi des outils plus élaborés qui permettront une mise en page assistée.

Imaginons l'index `entree` qui permet de gérer les entrées du dictionnaire que vous construisez. À la fin du livre, vous voulez afficher cette index sur trois colonnes. Vous allez donc implémenter la méthode `CustomIndexModule#printindex` (cf. page 82) `entree` dans le fichier `formater.rb`.

Elle ressemblera au code de la page suivante.

```
module CustomIndexModule
```

```

# Méthode appelée automatiquement si le code
# `(( index\\(entree) ))`
# est utilisé dans le texte
#
def printindex (cf. page 82)entree(pdf)

  # Les paramètres d'instanciation du multi colonnes
  params = {
    column_count: <nombre de colonnes>,
    width:        <largeur si autre que page complète>,
    gutter:       <gouttière si autre que valeur défaut>,
    lines_before: <nombre de lignes avant si autre que 1>,
    lines_after:  <nombre de lignes après si autre que 1>,
    space_before: <espace avant si nécessaire>,
    space_after:  <espace après si nécessaire>
  }
  # On instancie un texte multi-colonnes
  mc_block = Prawn4book::PdfBook::ColumnsBox.new(book, **params)
  # items contient la liste de toutes les entrées
  items.each_with_index do |item_id, occurrences, idx|
    # On traite les items pour obtenir le texte
    str = ...
    # On en fait une instance de paragraphe
    para = Prawn4book::PdfBook::NTextParagraph.new(
      book: book,
      raw_text: str,
      pindex:idx,
    )
    # On peut indiquer la source, pour les messages d'erreur
    # éventuels
    para.source = "Construction de l'index des entrées"
    # On peut supprimer l'indentation éventuelle
    para.indentation = 0
    # Et on injecte le paragraphe dans le bloc multi-colonnes
    mc_block << para
  end
  # On diminue la fonte à utiliser et on l'utilise
  fonte = Prawn4book::Fonte.dup_default
  fonte.size = 10
  pdf.font(fonte)
  # Il suffit maintenant d'imprimer ce bloc multi-colonnes
  mc_block.print(pdf)
  # Done!
end
end #/module

```

Formateurs

En mode expert, on a un accès complet à toutes les possibilités qu'offrent **Prawn-For-Book**, c'est-à-dire, n'ayons pas peur de le dire, *un monde infini, sans aucune limite*.

Pour se faciliter la vie, certaines méthodes propres permettent une implémentation plus rapide. C'est le cas de la méthode `Printer#pretty_render` qui, comme son nom l'indique, permet d'obtenir un bon rendu dans le livre sans effort (et, notamment, un traitement du texte sur les lignes de référence — cf. page)

Voici un code exemple :

```
# Par exemple dans ./formater.rb
def mon_helper(pdf)
  # On définit une fonte particulière
  mafonte = Prawn4book::Fonte.new(name:'Arial', size:8, style:
:normal)
  # On écrit le texte voulu dans le document.
  Printer.pretty_render(
    pdf:      pdf,
    fonte:    mafonte,
    text:     "Mon texte qui sera bien disposé [etc.]",
    options:  {left: 40, right: 80},
    owner:    nil,
  )
end
```

Le livre final (document PDF) contiendra :

Ci-dessous, je vais appeler la méthode `mon_helper` définie ci-dessus avec le code :

```
# { -mon_helper(pdf) } 13.
```

Mon texte qui sera bien disposé sur les lignes de référence malgré sa taille plus petite que le texte normal de ce mode d'emploi. C'est une police Arial, de taille 8, qui va se placer à 40 points de la marge gauche et à 80 points de la marge droite, car le `:left` des options a été mis à 40 et le `:right` a été mis à 80. Ça permet d'avoir un texte qui se place où on veut dans la page, plus serré que les marges.

2

¹³ Le « — » devant l'appel de la méthode permet de n'imprimer aucun retour de méthode. Rappel : sinon, c'est toujours le retour de l'appel qui est imprimé.

Modification de la hauteur de ligne

En tant qu'expert, vous pouvez modifier localement la hauteur de ligne (`line_height`) de deux façons.

Noter cependant que cette modification n'est pas à prendre à la légère puisqu'elle va bousculer l'harmonie des lignes du livre. Elle doit se faire en priorité sur une *belle page* (une page de gauche, paire) afin que les lignes des pages en vis à vis restent alignées et se

réserver à l'annexe du livre, par exemple, ou à la page des index, quand des tailles de police plus petites sont utilisées.

Modification de la hauteur de ligne dans le texte

Au fil du texte, la hauteur de ligne peut être redéfinie avec :

```
(( line_height(<nouvelle hauteur>, {fname:<police>,
  fsize:<taille>, fstyle:<style>})) ))
```

Par exemple :

```
(( line_height(10, {fname:'Garamond', fsize:9, fstyle:'normal'}))
))
```

Modification de la hauteur de ligne par formateur

On peut appeler une méthode définie dans `prawn4book.rb` par exemple :

```
# ./prawn4book.rb
module Prawn4book
  def redefinir_hauteur_de_ligne(pdf, valeur)
    pdf.line_height = valeur
    Fonte.default = ...
  end
end
```

Et dans le texte par exemple (mais cette méthode pourrait être appelée de n'importe où) :

Un texte quelconque.`#{redefinir_hauteur_de_ligne(10)}`

AVANT-PROPOS	13
FONCTIONNALITÉS	17
Les Recettes	19
Recette du livre	19
Recette de la collection	19
Définition de la fonte par défaut du texte	20
Définition des fontes (embarquées)	21
Éditeur / Maison d'édition	23
Le Texte du livre	25
Les types de paragraphes	25
Texte au format pseudo markdown	25
Formatages du texte	25
Introduction à la stylisation en ligne	26
Définition des titres	27
Aides & Assistants	30
Manuel et autres aides	30
Correspondances linguistiques	31
Les « Snippets »	31
Messages d'erreurs et notices	33
Afficher grille de référence et marges	34
Exportation du texte	34
Pagination	35
Les types de pagination	35
Types de numérotation	35
Numérotation des paragraphes	38
Suspension de la pagination	44
Le Texte en détail	49
Stylisation en ligne	49
Indentation des paragraphes	52
Gestion des tirets conditionnels	53
Placement sur une ligne quelconque	55
Les puces	56

Format du livre et des pages	69
Définition des marges	69
Section en multi-colonnes	72
Le Mode Expert	75
Description	76
Indication du contexte d'erreur	76
Référence à un paragraphe	76
Injection	78
Évaluation du code ruby	78
Méthodes d'helpers en mode expert	81
Parsing du paragraphe	82
Bibliographies en mode expert	83
Mode Multi-colonnes	83
Formateurs	85
Modification de la hauteur de ligne	85

Icare Éditions

Conception
Philippe Perret
(philippe.perret@icare-editions.fr)

Mise en page
Prawn-For-Book

Couverture
MM & PP

Correction & relecture
Marion Michel et Philippe Perret

Imprimé par Prawn-For-Book