Manuel de Prawn-for-book

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

Philippe Perret

Icare Éditions

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 XXX pages est auto-produit par « **Prawn-for-book** », c'est-à-dire qu'il est construit de façon *programmatique* 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

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 texte_intro_inline_stylingtexte/intro_inline_styling, 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 proprité, 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 chaine 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 virguler 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) :

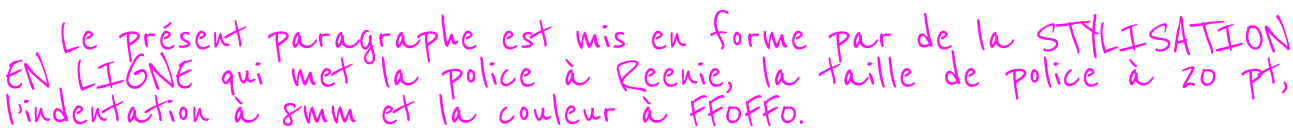
- **`size`** | Taille de la police du paragraphe suivant.
- **`font`** | Nom de la police à utiliser. Elle doit bien sûr être définie et embarquée (voir recette_definition_fontesrecette/definition_fontes).
- **`style`** | Le style à appliquer, parmi `normal`, `italic`, `bold`, `[:italic, :bold]` ou tout autre style défini explictement pour les fontes embarquées. Voir recette_definition_fontesrecette/definition_fontes.
- **`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 suplémentaire 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 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 :

U n t e x t e a v e c l e s l e t t r e s e s p a c é e s .
(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 texte_detail_stylisation_in_linestylisation en ligne (cf. page 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'identation du paragraphe

Il peut arriver, parfois, que l'identation 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 conditionnels [1] à l'aide de la marque dans le texte, à l'endroit où doit se trouver un tiret conditionnel.

[1] *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 anticonstitutionnellement 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.

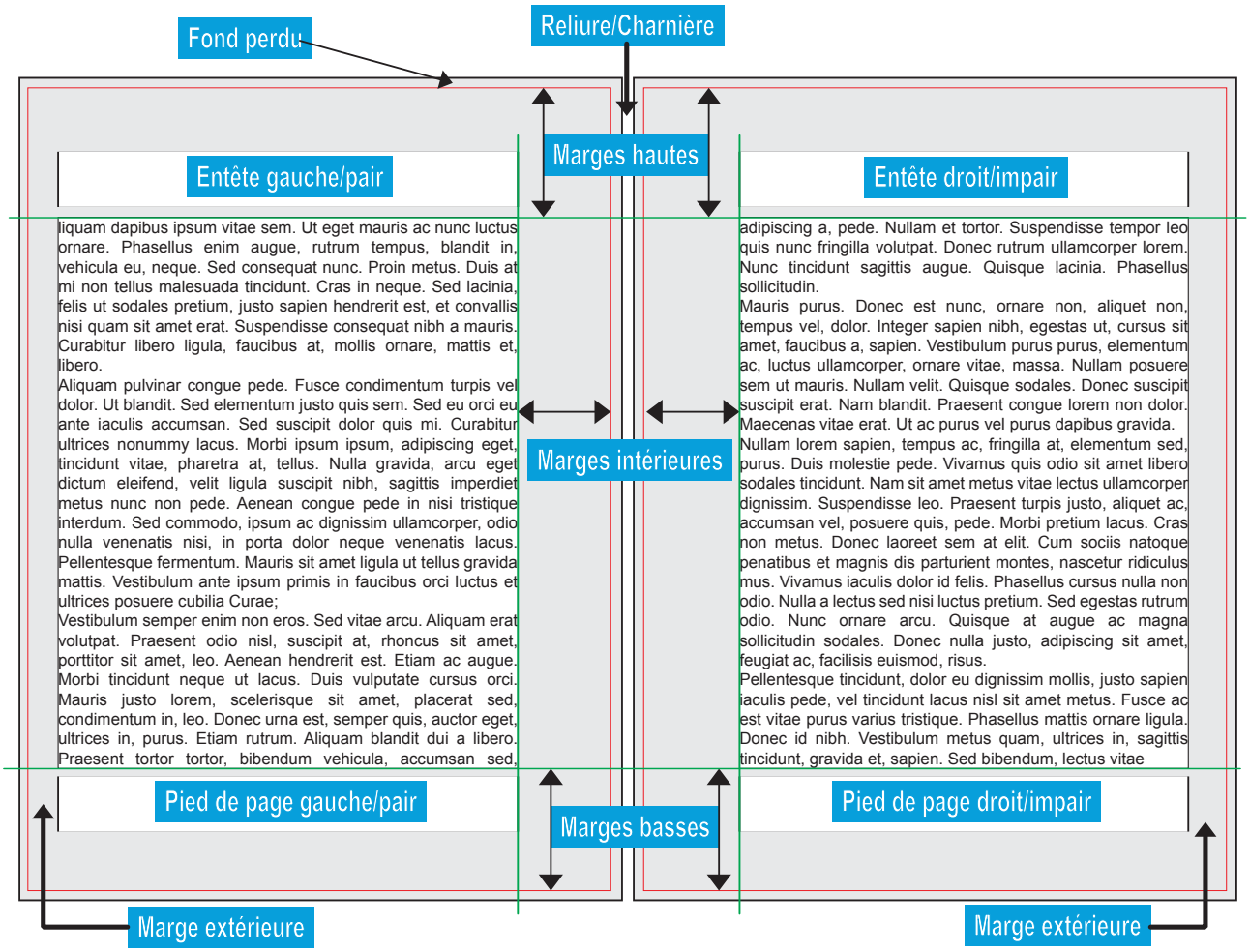
Le livre final (document PDF) contiendra :

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

Ce paragraphe est placé sur la dernière ligne.

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 goutière (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) ))` [2]

---

[2] *Vous remarquez ci-dessus l'utilisation d'une constante (cf. annexe_constantesannexe/constantes).*

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 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. 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. 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 texte en double colonnes qui ne devrait pas présenter de ligne vide de séparation ni au-dessus avec le texte avant ni au-dessous avec le texte après. Tous ces textes devraient être collés. Fin du texte en double colonnes.

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
```

Problèmes avec les multi colonnes

Cette section décrit les problèmes qui peuvent survenir avec les doubles ou multi colonnes.
Note : dans l'idéal, ce fichier devrait être supprimé car il sert surtout à reproduire les erreurs pour pouvoir les corriger dans l'application.

ET que le texte du livre contient…

Un texte au-dessus de la section en 2 colonnes avec des items.
((colonnes(2)))

Un premier item
Un deuxième item
Un troisième item
((colonnes(1)))
Un texte sous la partie en deux colonnes avec des items les uns sur les autres.
((new_page))
Un texte qui vient au-dessus de la section en trois colonnes.
((colonnes(3)))
(({align: :left}))
Un texte très très long pour qu'il tienne sur plusieurs colonnes. Pour voir comment le texte se répartira. Normalement, il ne devrait pas être justifié, mais aligné à gauche.
((colonnes(1)))
Un texte sous la partie en trois colonnes justifiée à gauche.
((new_page))
Un texte au-dessus de la section à deux colonnes.
((colonnes(2)))
(({align: :center}))
Ici, on devrait avoir un texte aligné au centre dans les deux colonnes qui ont été affectées à cette section.
((colonnes(1)))
Un texte sous la partie en deux colonnes justifiée au centre.
((new_page))
Un texte au-dessus de la section à deux colonnes avec changement d'alignement
((colonnes(2)))
Un premier paragraphe avec l'alignement par défaut, c'est-à-dire un texte justifié.
(({align: :right }))
Le paragraphe suivant (celui-ci) est aligné à droite et il est de moyenne longueur.
(({align: :left}))
On passe ensuite, ici, à un paragraphe aligné à gauche de la même longueur à peu près.
(({align: :center, size: 20, color: "FF0000"}))
Ce paragraphe est plus différent, avec une taille de police et une couleur différente. Il est également plus long.
((colonnes(1)))
Ce paragraphe se trouve en dessous de la section à plusieurs colonnes. Il sert à s'assurer qu'il n'y a pas d'espace entre la section multi-colonne et le texte qui suit.
((new_page))
Ce paragraphe se trouve au-dessus d'une section à 3 colonnes qui est espacé de 2 lignes de cette section.
((colonnes(3, {lines_before: 2, lines_after:4 })))
Une ligne
Une autre ligne
Une troisième ligne
((colonnes(1)))
Ce texte se trouve sous la section à 3 colonnes et à quatre lignes.

ALORS le livre contiendra…

Le texte qui doit apparaitre dans le livre.
Le retour de la construction du livre est :

```
c..par type : paragraph
```

```
par type : paragraph
par type : paragraph
Premier : "Un premier item"
......par type : pfbcode
par type : paragraph
Premier : "\n"
......par type : pfbcode
par type : paragraph
Premier : "\n"
......par type : paragraph
par type : paragraph
par type : paragraph
par type : pfbcode
par type : paragraph
par type : pfbcode
par type : paragraph
Premier : "Un premier paragraphe avec l’alignement par défaut,
c’est-à-dire un texte justifié."
.....par type : paragraph
par type : paragraph
par type : paragraph
Premier : "Une ligne"
.
[texte.pfb] 48 paragraphes instanciés et imprimés.
----------------------------------------
 Book PDF produit avec succès !
(in
Programmes/Prawn4book/Manuel/MANUEL_BUILDING/RealBooksCollection/f
ormat_precis_double_colonnes_erreurs/book.pdf)
48 paragraphes sur 6 pages traités en 0.11 s
```

Des items sur deux colonnes :

Un texte au-dessus de la section en 2 colonnes avec des items.

- Un premier item
- Un deuxième item
- Un troisième item

Un texte sous la partie en deux colonnes avec des items les uns sur les autres.

1

Une longue phrase sur trois colonnes :

Un texte qui vient au-dessus de la section en trois colonnes.

Un texte très très long pour qu'il tienne sur plusieurs colonnes. Pour voir comment le texte se répartira. Normalement, il ne devrait pas être justifié, mais aligné à gauche.

Un texte sous la partie en trois colonnes justifiée à gauche.

2

Un texte au-dessus de la section à deux colonnes.

Ici, on devrait avoir un texte aligné au centre dans les deux colonnes qui ont été affectées à cette section.

Un texte sous la partie en deux colonnes justifiée au centre.

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