



Manuel de Prawn-for-book

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

Philippe Perret

AVANT-PROPOS	11
FONCTIONNALITÉS	15
Généralités	17
Les Forces de <i>Prawn-For-Book</i>	17
Les deux fichiers de base	17
Les Recettes	19
Recette du livre	19
Recette de la collection	19
Définition des marges	21
Définition de la fonte par défaut	23
Éditeur / Maison d'édition	23
Le Texte du livre	24
Texte au format pseudo markdown	24
Les Pages spéciales	25
Les Types de pages spéciales	25
Table des matières	26
Page des crédits (Colophon)	29
Page de titre	32
Entêtes et Pieds de page	35
Description	35
Exemple complexe	38
Le Texte en détail	41
Les puces	41
Les Images	49
Insérer une image	50
Les Images flottantes	64
Bibliographies	76
Introduction	76
Liste des costumes	82
Le Mode Expert	84
Description	84
Indication du contexte d'erreur	84
Injection	84
Évaluation du code ruby	85

Bibliographies en mode expert	86
TUTORIEL	89
Premiers pas	90
ANNEXE	95
Marques markdown	96
Le format YAML de la recette	96
Les "Fontes-Strings"	97
Définition de la couleur	97
Rogner une image SVG	98
Synopsis de création	101
Liste exhaustive des fonctionnalités	101
Films cités	102
Articles de presse	102

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 108 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 Forces de *Prawn-For-Book*

Prawn-for-book possède de nombreuses forces et de nombreux atouts dont nous ne pouvons que donner un aperçu non exhaustif dans cette introduction.

- Sans configuration ou définitions, l'application produit un document PDF valide pour l'imprimerie professionnelle respectant toutes les normes et les habitudes en vigueur, avec par exemple toutes les lignes de texte alignées sur une grille de référence, avec une mise en page en contenant aucune veuve, aucune orpheline et aucune ligne de voleur, avec un contenu gérant les références croisées, les index et les bibliographies, avec un texte respectant toutes les consignes typographiques.
- Bien que **Prawn-For-Book** soit capable de produire de façon automatique ce document, l'application offre la possibilité de définir tous les éléments de façon très précise et très fine pour obtenir le rendu exact souhaité, grâce à une *recette* qui accompagne le texte et où peuvent être paramétrés tous les aspects du livre.
- Plus loin encore, **Prawn-For-Book** permet de travailler comme aucun autre logiciel sur **une collection entière**, grâce à un *fichier recette* partagé par tous les livres — chacun pouvant définir sa propre recette pour rectifier des aspects particuliers. De cette manière, on peut très simplement obtenir une collection cohérente partageant la même charte. On peut même obtenir des références croisées dynamiques entre les différents livres.

Le reste de ce manuel vous permettra de découvrir l'ensemble des fonctionnalités à votre disposition.

Les deux fichiers de base

Un livre **Prawn-For-Book** peut se définir, au minimum, par deux fichiers fonctionnant de paire :

- un fichier définissant le *texte du livre*, donc le **contenu**,
- un fichier définissant l'*aspect du livre*, appelé « recette ».

Le fichier contenu

Le fichier du texte porte impérativement le nom `texte.pfb.md` et contient l'ensemble du contenu du livre. Il peut aussi faire appel à des *helpers* qui produiront du contenu dynamique.

Par exemple, si c'est un dictionnaire, on peut imaginer que le livre travaille en parallèle avec une base de données contenant la définition des mots.

Les bibliographies sont aussi des exemples de contenus qui peuvent être gérés automatiquement, pour un confort plus grand, une agilité incroyable et une cohérence à toute épreuve.

Le fichier porte l'extension `pfb.md` qui est composée de `pfb` pour « Prawn For Book » et de `md`, extension naturelle du *markdown* qui est le langage qui a inspiré l'établissement du contenu. Comme nous le verrons, c'est cependant un *pseudo-markdown* qui est utilisé.

Le fichier recette

Il porte impérativement le nom `recipe.yaml` quand il concerne un livre et `recipe_collection.yaml` quand il concerne une collection (nous reviendrons en temps voulu sur cette distinction).

Comme l'indique son extension, ce fichier est au format `YAML`, un format très simple permettant de définir des données (cf. ci-dessous).

Signe distinctif

Le *signe distinctif* de **Prawn-For-Book**, utilisé pour tous les codes, est la double parenthèses :

```
(( . . . ))
```

Dès qu'un code doit être ajouté au texte, c'est souvent ce signe qui est utilisé. Vous pouvez le voir pour ce propre texte, où des `((line))` ont été utilisés pour sauter des lignes et un `(({align: :center}))` a été employé pour mettre le paragraphe suivant à la ligne.

Noter qu'il est impératif de laisser une espace à l'intérieur des parenthèses, de chaque côté. Dans le cas contraire, le code ne serait pas vu.

Si vous utilisez `### REF: annexe_package_sublime_text ###`, vous verrez le code se mettre en couleur, ce qui vous assurera que vous avez la bonne expression.

Description précédente obtenue à l'aide de :

Le **signe distinctif** de *****Prawn-For-Book*****, utilisé pour tous les codes, est la double parenthèses :

```
(( line ))
```

```
(( . . . ))
```

```
(( line ))
```

Dès qu'un code doit être ajouté au texte, c'est souvent ce signe qui est utilisé. Vous pouvez le voir pour ce propre texte, où des `((line))` ont été utilisés pour sauter des lignes et un `(({align: :center}))` a été employé pour mettre le paragraphe suivant à la ligne.

Noter qu'il est impératif de laisser une espace à l'intérieur des parenthèses, de chaque côté. Dans le cas contraire, le code ne serait pas vu.

Si vous utilisez `### REF: annexe_package_sublime_text ###`, vous verrez le code se mettre en couleur, ce qui vous assurera que vous avez la bonne expression.

Les Recettes

Recette du livre

La *recette du livre* 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. page 96).

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 page 96).

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

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

```
---
#<collection_data>
# (données de la collection)
```

```
collection_data:
  name: "Nom de la collection"
  short_name: "Nom raccourci (pour les messages)"
#</collection_data>
# ...
```

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 doubles, 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, contre le dos,
- une marge extérieure, qui comme son nom l'indique est tournée vers l'extérieur du livre, vers la tranche (attention, la tranche du livre, ça n'est pas la partie qui reprend le titre du livre et son auteur, qu'on appelle le « dos »).

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

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 `-display_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 le fichier recipe.yaml ou recipe_collection.yaml contient...

```
book_format:
  page:
    margins:
      top: 60mm
      bot: 70mm
      ext: 2mm
      int: 30mm
    show_margins: true
```

Le livre final (document PDF) contiendra :

Pour cette page, où les marges sont visibles, on illustre des marges de page à 60mm en haut, 70mm en bas, 30mm à l'intérieur et 2mm à l'extérieur.

Vous remarquez donc une immense marge en bas, une grande marge en haut, 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 la marge gauche et la marge droite, ce qui donne un rendu impeccable.

Définition de la fonte par défaut

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

```
book_format:
  text:
    default_font_and_style: "Times-Roman/"
    default_font_style: ""
    default_font_size: 20
```

Le livre final (document PDF) contiendra :

Pour écrire ce texte, nous avons ponctuellement modifié la police par défaut en utilisant la fonte Times-Roman avec une taille de 20 et le style .

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 rueCode Ville>"
  contact: "<contact@chez.lui>"
  url: "https://<url/site>"
  logo_path: "chemin/vers/logo"
```

Le Texte du livre

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 page 96, l'intégralité des marqueurs utilisables.

Les Pages spéciales

Les Types de pages spéciales

Les pages considérées comme *spéciales* dans **Prawn-For-Book** sont les suivantes.

- **Page de faux-titre.** Page contenant juste le titre et l'auteur. Par défaut, elle n'est pas placée dans un livre produit par **Prawn-For-Book**. Mettre `faux_titre` (ou `half_title_page`) à `true` dans la section `inserted_pages` de la recette pour l'imprimer.
 - **Page de garde.** Page vierge, après la couverture ou la page de faux titre si elle existe, qui permet de ne pas voir le titre par transparence. Par défaut, elle est placée dans un livre produit par **Prawn-For-Book**. Mettre `page_de_garde` (ou `endpage`) à `false` dans la section `inserted_pages` de la recette pour ne pas l'imprimer.
 - **Page de titre.** Page contenant toutes les informations de la couverture, le titre, le sous-titre éventuel, les auteurs, l'éditeur et son logo. Par défaut, elle est placée dans un livre produit par **Prawn-For-Book**. Mettre `page_de_titre` (ou `title_page`) à `false` dans la section `inserted_pages` de la recette pour ne pas l'imprimer.
 - **Page des crédits (colophon).** Page, à la fin du livre, contenant une sorte de *générique* du livre, avec la liste de tous les intervenants ayant contribué à la fabrication du livre (metteur en page, graphiste, concepteur de la couverture, etc.), les informations sur la maison d'édition, la date de parution, l'ISBN et tout autre renseignement utile. Par défaut, cette page n'est pas imprimée dans le livre. Mettre `page_credits` (ou `credits_page`) à `true` ou définir ses propriétés dans la section `inserted_pages` de la recette pour l'imprimer (cf. ci-dessous).
 - **Page d'index.** Page, vers la fin du livre, qui présente l'index de tous les mots... indexés.
- Nous allons aborder chaque page séparément. Ce que l'on peut retenir pour le moment, c'est que l'on détermine facilement dans la recette la présence ou non de ces pages dans la rubrique `inserted_pages` : (ci-dessous, ce sont les valeurs par défaut) :

Dans `recipe.yaml`

```
---
...
inserted_pages:
  page_de_garde:  true
  faux_titre:     false
  page_de_titre:
    title:
      font: "Times-Roman//18/000000"
      lines_before: 4
    copyright: "Tous droits réservés"
    logo:
      height: 10
  credits_page:  false
```

Table des matières

On imprime la table des matières à l'endroit voulu à l'aide de la marque :

`((tdm))` pour "T"able "D"es "M"atières

ou :

`((toc))` pour « T"able "Of » "C"ontent, la table des matières en anglais.

Noter que la table des matières, sauf indication contraire, se placera toujours sur une page paire, c'est-à-dire sur une fausse-page (page gauche). Cela part du principe qu'une table des matières fait en général au moins deux pages et qu'il est préférable de l'avoir sur une double page.

On peut cependant empêcher ce comportement en mettant la propriété `not_on_even` (« pas sur la paire » en anglais) à `true` dans la recette.

Table des matières en début d'ouvrage

Si on inscrit la table des matières en début d'ouvrage, il faut calculer le nombre de pages qu'elle va occuper et le définir dans la propriété `page_count` de la section `table_of_content` (« table des matières » en anglais). Ce nombre doit être pair et il vaut 2 par défaut.

Dans la pratique, estimez grossièrement la longueur de la table des matières en fonction de la taille de l'ouvrage et du nombre de titres et ajoutez 2 pour garder une certaine latitude. Ajustez le nombre final une fois l'ouvrage presque achevé pour supprimer les pages superflues (ou ajoutez-en de nouvelles si la table des matières « mange » sur le texte).

Table des matières en fin d'ouvrage

Si on inscrit la table des matières à la fin de l'ouvrage, il n'y a aucune précaution à prendre concernant le nombre de pages qu'elle couvrira.

Pour ce manuel, dont on trouve deux tables des matières, une en début d'ouvrage et l'autre en fin d'ouvrage. Seule la table des matières en début d'ouvrage a dû faire l'objet d'un calcul du nombre de pages.

Aspect de la table des matières

Comme les autres réglages, on peut définir précisément la table des matières dans les Recettes (page 19), dans la partie `table_of_content` (qui signifie « table des matières » en anglais).

On trouve ces propriétés :

- **pages_count** | Définit le nombre de pages réservées par la table des matières. Ce nombre doit impérativement être pair pour conserver l'agencement des *belles pages* (pages impaires) et des *fausses pages* (pages paires) dans le livre. Par défaut, on compte 2 pages pour la table des matières. Cette valeur vaut 2 par défaut. Noter que les pages supplémentaires (il y en aura toujours deux) peuvent aussi être ajoutées explicitement dans le livre par des `((new_page))`.
- **level_max** | Niveau maximum. Le niveau de titre qui sera affiché dans la table des matières. Par défaut, il est à 3, ce qui signifie que les titres jusqu'à 3 dièses seront affichés dans la table des matières (sauf exclusion).

- **title** | Le grand titre à utiliser pour la table des matières. Par défaut, en français, c'est « Table des matières ».
- **title_level**. Le niveau de titre pour ce grand titre. Par défaut, c'est 1.
- **no_title**. S'il ne faut pas imprimer de grand titre « Table des matières » (ou autre valeur de **title**), alors il faut mettre cette propriété à `true` (vrai).
- **font**. Pour la fonte générale (### REF: annexe_font_string*fonte string* ###)
- **number_font**. Fonte à utiliser pour les numéros de page (### REF: annexe_font_string*fonte string* ###).
- **number_size**. Si on ne veut changer que la taille du numéro (pas toute la fonte), on peut utiliser cette propriété.
- **numeroter**. Si `false` (faux), on ne numérote pas la table des matières (éviter).
- **lines_top** | Définit le nombre de lignes au-dessus du premier titre de chaque page de table des matières.
- **lines_bottom**. Nombre de lignes au-dessous du dernier titre de chaque page de table des matières.
- **line_height**. Hauteur de la ligne (entre chaque titre). Attention de ne pas donner une valeur plus petite que la taille des titres, sinon ils n'apparaîtront pas.
- **not_on_even** | Si `true` (vrai), on n'impose pas de commencer la table des matières sur une *fausse page* (à gauche). `false` par défaut.
- **vadjust_number**. Nombre de points-post-scripts pour ajuster verticalement le numéro de la page en face du titre (un nombre positif fait descendre le numéro, un nombre négatif le fait monter),
- **vadjust_line**. Nombre de points-ps pour ajuster verticalement la ligne pointillée d'alignement entre le titre et le numéro de page,
- **dash_line**. Les *experts* peuvent modifier la ligne pointillée en jouant sur cette propriété. Cf. plus bas, à propos de la ligne d'alignement.
- **levelX** où X va de 1 au niveau maximum de titre défini par `level_max`. Donc, par défaut, `level1`, `level2` et `level3`

Chaque niveau de titre peut définir :

- **font**. Sa fonte/style/taille/couleur,
- **size** ou seulement sa taille, en gardant la police par défaut de la table des matières,
- **indent**. Indentation du titre, donc son décalage à droite par rapport à la marge gauche.
- **number_size**. La taille du numéro (mais pour un meilleur aspect, il vaut mieux garder la taille générale).
- **caps**. La modification (casse) du titre. Les valeurs possibles sont `all-caps` (tout en majuscule), `all-min` (tout en minuscule), `title` (titre normal, en fonction de la langue du livre) ou `none` pour le laisser tel quel.
- **number_font**. La police fonte/style/taille/couleur à utiliser pour les numéros de ce niveau de titre (mais il vaut mieux une grande cohérence avec l'ensemble et éviter de la définir).

Exclusion de titres dans la table des matières

On peut exclure de la table des matières des titres du niveau voulu (`level_max`) en ajoutant `{no_toc}` à leur titre, soit à la fin soit après les dièses. Par exemple :

```
### {no_toc} Titre hors TdM
```

ou :

```
## Titre hors TdM {no_toc}
```

Vous pouvez vérifier que le titre dans le texte en exemple ci-dessous ne sera pas imprimé dans la table des matières.

(rappel : « *toc* » signifie « *table of content* », c'est-à-dire « *table des matières* » en anglais — on peut aussi utiliser `{no_tdm}`)

Numérotation de la table des matières

Le « numéro de page » utilisé dans la table des matières dépend directement de la pagination utilisée pour le livre. C'est le numéro de page seule si la pagination est à pages, c'est le numéro de paragraphe si la pagination est à paragraphes et c'est à page et paragraphe si la pagination est hybrid.

Ligne d'alignement de la table des matières

La *ligne d'alignement* est une ligne, souvent pointillée, qui relie le titre (à gauche) au numéro de page (à droite), et permet de mieux faire correspondre visuellement titre et numéro de page.

Elle est définie dans la recette à l'aide de la propriété `dash_line`.

`dash_line` est une table définissant `{:length, :space, :color}` où `:length` est la longueur du tiret (1 pour un point), `:space` est l'espace entre deux tirets et `:color` est la couleur hexadécimale du trait.

Astuce : on peut obtenir une ligne très fine, très discrète, en mettant `dash_line` à `{length:1, space:0, color:"DDDDDD"}`. Noter que c'est le `space: 0` qui, en ne laissant aucun espace entre les points, génère une ligne.

Propriétés qu'on peut trouver dans la recette

```
---
...
table_of_content:
  # Voir ci-dessus les explications de chaque propriété
  # - Titre -
  title: "Table des matières"
  title_level: 1
  no_title: false
  # - Contenu -
  level_max: 3
  numeroter: true
  # - Aspect -
  lines_top: 4
  lines_bottom: 4
  font: "<font>/<style>/<size>/<color>"
  number_font: "<font>/<style>/<size>/<color>"
  number_size: 10
  line_height: 14
  vadjust_line: 0      # ligne pointillée
  vadjust_number: 0    # numéro page
  dash_line: {length: 1, space: 3}
  # - Aspect des titres par niveau -
  level1:
    font: "<font>/<style>/<size>/<color>"
    size: 12
    number_size: null # => même que défaut
    caps: all-caps
    indent: 0
  level2:
```

```
...
  indent: 1cm
level3:
  ...
  indent: 15mm
```

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

{no_toc} TITRE HORS TDM

Vous pouvez vérifier que ce titre ne sera placé ni dans la table des matières de début de livre ni dans celle de fin de livre.

Le livre final (document PDF) contiendra :

TITRE HORS TDM

Vous pouvez vérifier que ce titre ne sera placé ni dans la table des matières de début de livre ni dans celle de fin de livre.

Page des crédits (Colophon)

On appelle *page des crédits* ou *colophon* la page placée à la fin du livre qui donne toutes les informations sur le livre, aussi bien au niveau de l'éditeur ou la maison d'éditions (son nom, son adresse, son email) qu'au niveau de toutes les personnes qui ont contribué à conception du livre, rédacteur ou rédactrice, metteur ou metteuse en page, graphiste, concepteur, directeur ou directrice de collection, en passant par d'autres informations comme le numéro ISBN du livre et sa date de parution ou les remerciements.

Si vous êtes de niveau *Expert*, vous pouvez bien entendu mettre cette page en forme de façon tout à fait personnalisée dans le détail, en faisant appel aux éléments de la recette (pour rappel, dans les modules ruby, cela s'obtient en faisant appel à la méthode `book.recipe` qui permet d'atteindre toutes les valeurs de la recette, et donc les informations de cette page.

Mais sans être *expert*, on peut obtenir une mise en page tout à fait correcte et professionnelle ici encore grâce aux comportements par défaut. Sans rien préciser d'autre que les informations minimales requises pour le livre ou la collection, la page des infos sera placée à la fin du livre si `pages_speciales: credits_page:` est à `true` dans la recette.

IMPRESSION DE LA PAGE DE CRÉDITS

La première chose à faire pour s'assurer que la page de crédits sera imprimée est de mettre la valeur `inserted_pages: credits_page:` à `true` (vrai).

Ensuite, il faut s'assurer que les valeurs minimales soient fournies. Vous pouvez lancer une première fois la construction du livre pour connaître les informations manquantes.

INFORMATIONS REQUISES

Toutes les informations requises pour la page des informations se trouvent dans les données `book_making` (« fabrication du livre » en anglais) et `credits_page` de la recette du livre et/ou de la collection.

Toutes les données de `book_making` — le ou les concepteurs du livre, le ou les rédacteurs, le ou les metteurs en page, graphiste, concepteurs de la couverture ou correcteurs — comprennent deux informations :

- **patro** | Pour le patronyme, seul ou une liste. Le patronyme s'écrit toujours avec la même convention : le prénom en minuscule avec capitale au début, le nom tout en capitales. S'il y en a plusieurs, on les met entre crochets (cf. l'exemple ci-dessous). Par défaut, les noms seront corrigés pour être corrects au niveau de la typographie (« Philippe PERRET » dans la donnée recette s'affichera « Philippe Perret »). Pour que le nom reste identique, il suffit de le faire précéder d'un signe égal).
- **mail** | Le mail du **patro** ci-dessus. S'il y a plusieurs personnes, on indique les mails dans le même ordre, entre crochets (comme ci-dessous).

La page des crédits contient aussi les informations sur l'éditeur ou la maison d'édition, qui doivent se trouver dans la section `publisher` de la recette. La seule information requise est le nom de l'éditeur (`publisher: name:`).

Vous trouverez ci-dessous, dans l'exemple de recette, toutes les informations utilisables dans la page de crédits.

DISPOSITION

Les crédits du colophon peuvent se placer en haut, en bas ou au milieu de la page. On définit cette position grâce à la propriété `credits_page: disposition:` qui peut prendre les valeurs "distribute" (ou "middle", milieu), "top" (haut de page) ou "bottom" (bas de page).

La valeur par défaut des "distribute".

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

```
---
inserted_pages:
  credits_page: true # pour qu'elle soit imprimée
book_data:
  isbn: ...
  url: "https://url/du/livre"
book_making:
  conception:
    patro: "Prenom NOM"
    mail: "mail@chez.lui"
  writing:
    patro: "Prenom NOM"
    mail: "mail@chez.lui"
  page_design: # Mise en page
    patro: ["Prénom NOM", "Prénom NOM"]
    mail: ["premier@chez.lui", "deuxieme@chez.lui"]
  cover: # Couverture
```

```

    patro: "=MM & PP" # '=' => non transformé
    mail: null
correction:
    patro: ...
    mail: ...
acknowledgements: # remerciements
    patro: ...
publisher:
    name: "<nom éditeur ou édition>"
    address: <Rue et numéronLieu ditnVille et code postal
    url: "<url du site internet>"
    mail: "<mail@editeur.fr>"
    contact: "<mail contact>"
    siret: "<numéro siret>"
credits_page:
    disposition: "distribute" # ou "bottom", "top"
    label:
        font: "fonte/style/taille/couleur"
    value:
        font: "fonte/style/taille/couleur"
printing:
    name: "à la demande" # p.e. pour KDP
    lieu: null

```

Extrait de la recette actuelle

```

---
# ...
book_making:
    conception:
        patro: Philippe PERRET
        mail: philippe.perret@icare-editions.fr
    writing:
        patro:
        mail:
    page_design:
        patro: Prawn-For-Book
        mail:
    cover:
        patro: "=MM & PP"
        mail:
    correction:
        patro: Marion MICHEL
        mail:
    printing:
        name: Prawn-For-Book
        lieu:
credits_page:
inserted_pages:
    page_de_garde: true
    faux_titre: false
    page_de_titre:

```

```

title:
  font: Numito/normal/30/FF0000
  line: 15
subtitle:
  font: Numito/italic/16/007700
  line: 16
author:
  font: Numito/normal/18/000077
  line: 22
collection:
  font: Times-Roman//12/000000
  line: 1
publisher:
  font: Numito/normal/12/FF8000
  line: 2
  logo:
    height: 40
    line: 3
copyright: Tous droits réservés
credits_page:
  disposition: distribute

```

Page de titre

La *page de titre* présente un grand nombre d'informations au lecteur et notamment le titre, l'auteur et l'éditeur du livre.

Ses données sont définies dans la section `inserted_pages` de la recette du livre ou de la collection (cf. page 19) grâce à la propriété `page_de_titre` (ou `title_page`).

Dans sa version la plus simple, utilisant les valeurs par défaut, on aura juste à indiquer :

```

inserted_pages:
  page_de_titre: true

```

... pour que la page de titre soit affichée en début de livre.

(mettre `false` — "faux" — au lieu de `true` — "vrai" — pour que la page de titre ne soit pas affichée)

Mais comme tout autre élément de **Prawn-For-Book**, on peut définir la page de titre de façon plus précise. Trouvez ci-dessous les propriétés définissables et retrouvez-les dans l'exemple de recette plus bas.

Vous pourrez aussi trouver, tout en bas de cette section, la recette utilisée pour produire la page de titre un peu trop bariolée (pour cette illustration) de ce manuel, dans les premières pages.

PROPRIÉTÉS DE LA PAGE DE TITRE

Ci-dessous, les sous-propriétés `font` sont des « font-string » (cf. page 96) et les propriétés `line` sont les lignes de référence sur lesquelles doivent être imprimés les éléments. Pour les voir sur les pages, vous pouvez demander la construction du livre avec l'option `-grid` qui les fera apparaître sur les pages.

Noter qu'une propriété ne sera imprimée que si elle est définie dans cette section `inserted_pages: page_de_titre:`.

- **title** | Définition de l'aspect du *titre du livre* sur la page de titre. Définit les propriétés `font` et `line` (ligne sur laquelle imprimer le titre).
- **subtitle** | Définition de l'aspect du *sous-titre du livre* s'il en possède un. Définit les propriétés `font` et `line`.
- **author** | Définit l'aspect de l'affichage de l'*auteur* (ou des auteurs) par les propriétés `font` et `line` (ligne sur laquelle imprimer l'auteur)
- **collection** | Définit l'aspect du titre de la collection à l'aide des propriétés `font` et `line`.
- **publisher** | Définit l'aspect de la maison d'édition ou de l'éditeur (« publisher » = éditeur). En plus des propriétés `font` et `line`, on trouve la propriété `logo` qui doit définir la taille en hauteur (`height`) et la ligne (`line`) du logo ¹.

¹ Ce logo doit être défini dans la section `publisher` de la recette (cf. page 23).

Propriétés du fichier recette

```
---
inserted_pages:
  page_de_titre:
    title:
      font: "fonte/style/taille/couleur"
      line: 4
    subtitle:
      # idem
    author:
      # idem
    collection:
      # idem
    publisher:
      # idem
      logo:
        height: <hauteur>
        line: <ligne>
    copyright: "<mention du copyright>"
```

Extrait de la recette actuelle

```
---
# ...
inserted_pages:
  page_de_garde: true
  faux_titre: false
  page_de_titre:
    title:
      font: Numito/normal/30/FF0000
      line: 15
    subtitle:
      font: Numito/italic/16/007700
      line: 16
```

author:
 font: Numito/normal/18/000077
 line: 22
collection:
 font: Times-Roman//12/000000
 line: 1
publisher:
 font: Numito/normal/12/FF8000
 line: 2
 logo:
 height: 40
 line: 3
 copyright: Tous droits réservés
credits_page:
 disposition: distribute

Entêtes et Pieds de page

Description

Comme pour les autres éléments, on pourra laisser les **entêtes** et **pieds de page** par défaut, ce qui signifiera n'afficher que les numéros de pages — sur les pages adéquates —, ou au contraire on pourra définir des entêtes et pieds de page complexes et adaptés au contenu pour une navigation optimum.

Comme pour les autres éléments de **Prawn-For-Book**, les entêtes et pieds de page par défaut sont conçus pour être directement « professionnels ». C'est-à-dire que la numérotation est intelligente, elle ne numérote pas bêtement toutes les pages de la première à la dernière. Seules sont numérotées les pages qui le sont dans un livre imprimé. Sont soigneusement évitées les pages vides, les pages de titre ou les pages spéciales comme table des matières (page 25) ou page des crédits (Colophon (page 29)).

Les pages suivantes vont définir les différents entête et pieds de page que l'on peut définir, en présentant dans la page le code utilisé et le résultat dans les entêtes et/ou les pieds de page.

Nomenclature des entêtes et pieds de page

Comme pour les autres parties, nous définissons ici la liste des termes qui seront rigoureusement utilisés dans cette partie et propres aux entêtes et pieds de page.

- **Entête.** C'est la partie, en haut de page, au-dessus du texte principal de la page, qui contient le plus souvent le titre courant, qui permet de naviguer plus facilement dans les chapitres du livre.
- **Header.** *Entête*, en anglais.
- **Pied de page.** C'est la partie, en bas de page, sous le texte principal de la page, qui contient le plus souvent le *numéro de page*.
- **Footer.** *Pied de page*, anglais.
- **Portion.** Nous appelons « portion » l'un des 6 tiers de page qui constituent une *entête* ou un *pied de page*. Il y en a 6 parce qu'on travaille ici en double page. On trouve la *portion gauche*, la *portion droite* et la *portion centrale* de la page gauche et les mêmes portions sur la page droite.
- **Disposition.** Une *disposition* décrit le contenu des pieds et page et entête d'une ou plusieurs pages. Cela revient à définir les 6 portions de l'entête et les 6 portions du pied de page.

TROIS PORTIONS DE PAGE

Un *entête* ou un *pied de page* est un espace de page qui contient jusqu'à trois portions, trois « tiers » de page, une portion gauche, une portion droite et une portion centrale. On répartit les éléments dans ces trois portions. Les trois portions peuvent être différentes entre page gauche et page droite.

ENTÊTES ET PIEDS DE PAGE

Puisqu'on travaille les entêtes et pieds de page en double page, on peut donc définir jusqu'à 6 portions qu'on peut représenter la manière suivante, en considérant que || représente la reliure du livre :

| gauche | centre | droit || gauche | centre | droit |

DÉFINITION DES ENTÊTES ET PIEDS DE PAGE

Les *entêtes* et *pieds de page*, comme pour le reste, se définissent dans les Recettes (page 19) du livre ou de la collection, dans une section qui s'appelle, on ne s'en étonnera pas : `headers_footers`.

On peut définir autant de *dispositions* que l'on veut, même s'il est conseillé, toujours, de rester le plus sobre possible. On peut se contenter, pour un résultat optimum, d'une *disposition* pour le corps du livre, son contenu principal, et une *disposition* pour les annexes si elles existent.

La propriété principale de la disposition est la valeur de son header (son « entête ») et son footer (« pied de page »). Par défaut, les valeurs sont :

- header : "| x | x | x || x | x | x |"
- footer : "| -NUM | x | x || x | x | NUM - |"

On voit clairement la reliure représentée par || et de chaque côté les trois portions.

Les x signifient qu'on ne met rien dans ces portions.

Le NUM indique l'endroit où l'on va marquer le numéro de la page. Voir ci-dessous les *éléments* qui peuvent composer une disposition.

Le tiret (moins) qui précède ou suit NUM indique l'alignement dans la portion. Le trait avant (-NUM) signifie que le numéro sera aligné à gauche, le trait après (NUM-) signifie que le numéro sera aligné à droite. Pour le centrer, on aurait rien mit car l'élément est aligné au centre par défaut.

Mais en fait, la vraie disposition est plus simple, car il est inutile de définir toutes les portions si elles sont vides. La vraie disposition par défaut dans **Prawn-For-Book** est :

- header : "| x || x |"
- footer : "| -NUM || NUM - |"

LES ÉLÉMENTS

Les éléments qui peuvent être utilisés dans les entêtes et les pieds de page sont illimités en mode expert (### REF: expert_header_footer ###). Pour le commun des mortels, ils se limitent — ce qui est déjà largement suffisant — aux titres courants jusqu'au niveau 5 (en majuscules, en minuscules, ou format titre), au numéro de la page ainsi qu'au nombre total de page.

- **NUM** pour le numéro de page,
- **TIT1** pour le titre de niveau 1 courant, en majuscules,
- **tit1** pour le titre de niveau 1 courant, en minuscules,
- **Tit1** pour le titre de niveau 1 courant, en format titre,
- **TIT2** (et **tit2**, **Tit2**) pour le titre de niveau 2 courant,
- **TIT3** (et **tit3**, **Tit3**) pour le titre de niveau 3 courant,
- **TIT4** (et **tit4**, **Tit4**) pour le titre de niveau 4 courant,
- **TIT5** (et **tit5**, **Tit5**) pour le titre de niveau 5 courant,
- **TOT** pour le nombre total de pages,
- **#{code}** pour un code ruby quelconque à évaluer à la volée, par exemple un numéro de version ou des auteurs,

AJUSTEMENT

Noter bien que l'*ajustement vertical* effectué avec `header_vadjust` et `footer_vadjust` fonctionne différemment. Dans les deux cas, il définit l'éloignement *par rapport au texte principal de la page*. Ainsi, pour `header_vadjust` (l'entête), une valeur positive fera monter les portions alors que pour `footer_vadjust` (le pied de page), une valeur positive les fera descendre.

FONTES

Bien qu'on puisse définir les polices très précisément pour chaque élément, il est conseillé de ne pas trop les différencier. La sobriété est toujours bonne conseillère, en matière de mise en page.

C'est la raison pour laquelle, par défaut, on ne peut définir que la police générale des deux *entête* et *pied de page*, à l'aide de la propriété `font` (qui est une `### REF: annexe_font_string"fonte-string" ###`), et/ou les propriétés `header_font` (fonte pour les entêtes) et `footer_font` (fonte pour les pieds de page).

Vous pourrez trouver dans les exemples suivants l'utilisation de ces propriétés.

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

```
---
# ...
# Définit le début de la définition des entêtes et pieds de
# page
headers_footers:
  # Définition des dispositions
  dispositions:
    # Pour supprimer la disposition par défaut, il suffit
    # de faire :
    default: null
    # Définition d'une disposition
    ma_disposition:
      name: "Nom humain juste pour mémoire"
      # Fonte particulière pour cette disposition en
      # particulier :
      font: "<font name>/<font style>/<font size>/<font color>"
      pages: <première page>-<dernière page>
      header: | x | x | x || x | x | x |
      header_font: "<fonte>/<style>/<size>/<color>"
      header_vadjust: <ajustement vertical entête>
      header_hadjust: <ajustement horizontal entete>
      footer_vadjust: <ajustement vertical pied>
      footer_vadjust: <ajustement vertical pied>
      footer_font: "<fonte>/<style>/<size>/<color>"
      footer: | x | x | x || x | x | x |
  - name: "Autre disposition"
    font: ...
    etc.
```

ENTÊTES ET PIEDS DE PAGE

Exemple complexe

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

```
---
headers_footers:
  dispositions:
    exemple1:
      name: "Exemple headers/footers complexe pour manuel"
      font: "Courier/regular/12/FF0000"
      pages: "36-41"
      header: "| -TIT3      || -v4.452 | TIT4- |"
      footer: "| x | NUM || NUM/TOT | PhP |"
      header_font: "Reenie/normal/20/009900"
```

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

((new_page))

Une première page pour voir l'entête et le pied de page définis (ça se poursuit sur l'autre page).

((new_page))

Une seconde page qui doit permettre aussi de voir l'entête et le pied de page sur les deux pages, car on ne sait pas, **a priori**, où va se retrouver cette page puisque le manuel est autoproduit par ****Prawn-For-Book**** lui-même !

Donc cette page que vous lisez peut se trouver en "fausse page", c'est-à-dire à gauche du livre, en page paire, aussi bien qu'en "belle page", c'est-à-dire à droite du livre, en page impaire.

Ainsi, dans tous les cas vous devriez pouvoir obtenir une nouvelle page contenant l'entête et le pied de page voulu.

Donc, à la différence des pages par défaut, on devrait trouver ici, à gauche, le numéro de page au milieu, alors qu'il est à gauche dans le pied de page droit, avec le nombre total de pages dans le manuel.

On doit trouver en à gauche de la page gauche le titre principal courant, donc le chapitre, qui s'intitule "Entêtes et Pieds de pages".

Et on doit trouver sur la page droite, en entête, le numéro de version "4.452" à gauche et au milieu le sous-titre du chapitre donc "Exemple complexe".

Le livre final (document PDF) contiendra :

v4.452

Une première page pour voir l'entête et le pied de page définis (ça se poursuit sur l'autre page).

ENTÊTES ET PIEDS DE PAGE

Une seconde page qui doit permettre aussi de voir l'entête et le pied de page sur les deux pages, car on ne sait pas, *a priori*, où va se retrouver cette page puisque le manuel est autoproduit par **Prawn-For-Book** lui-même !

Donc cette page que vous lisez peut se trouver en « fausse page », c'est-à-dire à gauche du livre, en page paire, aussi bien qu'en « belle page », c'est-à-dire à droite du livre, en page impaire.

Ainsi, dans tous les cas vous devriez pouvoir obtenir une nouvelle page contenant l'entête et le pied de page voulu.

Donc, à la différence des pages par défaut, on devrait trouver ici, à gauche, le numéro de page au milieu, alors qu'il est à gauche dans le pied de page droit, avec le nombre total de pages dans le manuel.

On doit trouver en à gauche de la page gauche le titre principal courant, donc le chapitre, qui s'intitule « Entêtes et Pieds de pages ».

Et on doit trouver sur la page droite, en entête, le numéro de version « 4.452 » à gauche et au milieu le sous-titre du chapitre donc « Exemple complexe ».

Le Texte en détail

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

On peut définir une autre puce dans la page 19 ou la page 19, dans la partie `text :` de `book_format :`.

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

```
---
book_format:
  text:
    puce:
      text: :losange
      vadjust: 2
      hadjust: 4
      left: 20mm
      size: 18
```

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

* Ceci est une puce losange de 18 points, remontée de 2 points, déplacée à droite de 4, et écartée du texte de 20 millimètres,
 * Le second item de liste est identique,
 * Ainsi que le troisième.

Le livre final (document PDF) contiendra :

- ◇ Ceci est une puce losange de 18 points, remontée de 2 points, déplacée à droite de 4, et écartée du texte de 20 millimètres,
 - ◇ Le second item de liste est identique,
 - ◇ Ainsi que le troisième.
-

Puce losange noir

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

```
#<book_format>
book_format:
  text:
    puce:
      text: :black_losange
      size: 14
      vadjust: -2
#</book_format>
```

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

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

Le livre final (document PDF) contiendra :

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

Puce carrée

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

```
#<book_format>
book_format:
  text:
    puce:
      text: :square
      size: 20
      vadjust: 2
#</book_format>
```

Si le fichier « `texte.pfb.md` » 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.

Le livre final (document PDF) 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.
-

Puce carrée noire

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

```
#<book_format>
book_format:
  text:
    puce:
      text: :black_square
      size: 20
      vadjust: 2
#</book_format>
```

Si le fichier « `texte.pfb.md` » 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.

Le livre final (document PDF) 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.
-

Puce ronde

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

```
#<book_format>
book_format:
  text:
    puce:
      text: :bullet
      size: 20
      vadjust: 1
#</book_format>
```

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

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

Le livre final (document PDF) contiendra :

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

Puce ronde noire

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

```
#<book_format>
book_format:
  text:
    puce:
      text: :black_bullet
      size: 30
      vadjust: 6
      left: 6.5mm
#</book_format>
```

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

- * Ceci est une puce `:black_bullet` de 30 points, remontée de 6 points, avec les autres paramètres laissés intacts,
- * Le second item identique,
- * Le troisième.

Le livre final (document PDF) contiendra :

- Ceci est une puce `:black_bullet` de 30 points, remontée de 6 points, avec les autres paramètres laissés intacts,
 - Le second item identique,
 - Le troisième.
-

Puce doigt tendu

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

```
#<book_format>
book_format:
  text:
    puce:
```

```

      text: :finger
      size: 20
      vadjust: 1
      left: 6.5mm
#</book_format>

```

Si le fichier « texte.pfb.md » 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.

Le livre final (document PDF) 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.
-

Puce doigt tendu noir

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

```

#<book_format>
book_format:
  text:
    puce:
      text: :black_finger
      size: 20
      vadjust: 1
      left: 7mm
#</book_format>

```

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

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

Le livre final (document PDF) contiendra :

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

Puce gros tiret

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

```
#<book_format>
book_format:
  text:
    puce:
      text: :hyphen
      size: 100
      vadjust: 65
      left: 20mm
#</book_format>
```

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

- * Ceci est une puce `:hyphen` de 100 points, remontée de 65 points, texte décalé de 20mm, avec les autres paramètres laissés intacts,
- * Le second item identique,
- * Le troisième.

Le livre final (document PDF) contiendra :

- Ceci est une puce `:hyphen` de 100 points, remontée de 65 points, texte décalé de 20mm, avec les autres paramètres laissés intacts,
 - Le second item identique,
 - Le troisième.
-

Puce personnalisées

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

```
#<book_format>
book_format:
  text:
    puce:
      text: 'images/custom_bullet.png'
      size: 14
      vadjust: -2
      left: 8mm
#</book_format>
```

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

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

Le livre final (document PDF) contiendra :

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

Les Images

Insérer une image

Pour insérer une image dans le flux du livre, on utilise le code pseudo-markdown **![chemin/vers/image]**. Si des données sont à passer, on utilise **![vers/image](<data>)** où <data> est une **### REF: table_ruby ###**.

Noter que ce code doit obligatoirement se trouver sur une ligne seule.

La suite va présenter toutes les propriétés utilisables avec les images, mais les pages suivantes reprendront toutes ces propriétés pour en donner des exemples concrets et illustrés.

CHEMIN D'ACCÈS À L'IMAGE

Le chemin d'accès à l'image peut se donner...

- relativement au dossier du livre,
- relativement au dossier 'images' du livre (s'il existe),
- relativement au dossier de la collection s'il y a collection,
- relativement au dossier 'images' de la collection s'il y a collection.

Les formats (jpg, tiff, png, svg)

POSITIONNEMENT DE L'IMAGE

Par défaut, l'image est placée au centre. Pour la mettre à droite ou à gauche, utiliser respectivement `align: right`, `align: left` et/ou la définition de `left` (décalage avec la marge gauche) et `right` (décalage avec la marge droite), comme dans les exemples ci-dessous.

Par défaut, l'image s'insère de façon fluide et naturelle avec le texte qui la précède et qui la suit. Mais on peut définir l'espacement de façon très précise avec les propriétés `space_before` (« espace avant ») et `space_after` (« espace après »), comme dans les exemples suivant (voir l'exemple 18, par exemple, à la page page 62).

Pour un placement optimum, on peut ajuster la position de l'image verticalement à l'aide de `vadjust` (« ajustement vertical »). L'ajustement horizontal, bien entendu, peut se faire avec `right` et `left`. Bien sûr, il est raisonnable de ne faire ces ajustements qu'à la dernière minute, lorsque le livre est presque finalisé.

TAILLE DE L'IMAGE

La taille de l'image peut être définie avec `width:` (largeur), `height:` (hauteur) et/ou `scale:` (échelle).

Notez que la largeur sera ramenée par défaut à la largeur de la page du livre si elle est supérieure, et la hauteur de la page si sa hauteur l'excède. Mais si l'on sait ce que l'on veut obtenir et qu'on désire tout contrôler soi-même, on peut ajouter l'option `no_resize: true` qui empêchera le redimensionnement naturel (à vos risques et péril comme dans l'exemple 19 à la page page 63).

LÉGENDE DE L'IMAGE

Une légende peut être définie en renseignant la donnée `legend`: "La légende de l'image". Par défaut, cette légende sera inscrite avec la même police que la police courante, avec une taille de une unité inférieure, en noir et en italique.

Mais on peut régler tous ces aspects :

- soit pour tout le livre (ou toute la collection) dans le fichier recette, dans `book_format > images > legend` (avec les propriétés `font`, `size`, `style` et `color`) — comme ce sera illustré dans la partie page 73.
- soit de façon ponctuelle pour une seule image/légende en définissant dans les données de l'image les propriétés `legend_font` (nom de la police), `legend_style` (style de la police, par exemple `:regular`), `legend_size` (taille de la police) et `legend_color` (couleur de la légende).

IMAGE SVG

Les images SVG sont incontestablement les meilleures images pour l'impression dans le sens où elles conservent leurs qualités quelle que soit la taille adoptée. On peut le voir sur les exemples ci-dessous. Elles sont donc à privilégier.

ROGNER UNE IMAGE SVG

Pour savoir comment rogner une image svg, voir page 98.

EXEMPLES

Les exemples ci-dessous sont tous construits à partir du code exact donné en exemple. Vous pouvez donc lui faire pleinement confiance.

Aspect des images en fonction du code

EXEMPLE 1 • Une image trop large sera réduite à la taille du livre.

! [images/exemples/plus_large.jpg]

Adagio

EXEMPLE 2 • La même image, mais avec une légende, insérée (notez qu'ici on n'a pas indiqué images dans le chemin d'accès, puisque ce dossier est implicite).

! [exemples/plus_large.jpg] (legend: "Image plus grande réduite en largeur")

Adagio

Image plus grande réduite en largeur

EXEMPLE 3 • Une image PNG trop haute, *sans légende*, qui sera réduite à la taille d'une page du livre et mise sur la page suivante.
![exemples/plus_haute.png]

NOTTURNO
für das Pianoforte
von
RIEDRICH CHOPIN.
Nachgelassenes Werk. (Op. 72. N^o 1)



EXEMPLE 4 • Une image PNG trop haute, avec *légende*, qui sera réduite à la taille d'une page du livre et mise sur la page suivante.

`![exemples/plus_haute.png](legend:"Image plus haute réduite
et placée sur la page suivante")`

NOTTURNO

für das Pianoforte
von
RIEDRICH CHOPIN.
Nachgelassenes Werk. (Op. 72. N^o 1)

♩. * ♩. * ♩. * ♩.

cresc. * ♩. * ♩. * ♩. * ♩. *dim.*

poco a poco cresc. ♩. * ♩. * ♩. * ♩. * ♩.

♩. * ♩. * ♩. * ♩. * ♩. *

* ♩. * ♩. * ♩. *

C. XIII. 31.

Image plus haute réduite et
placée sur la page suivante

EXEMPLE 5 • Une image PNG trop haute, dont on réduit explicitement la hauteur.

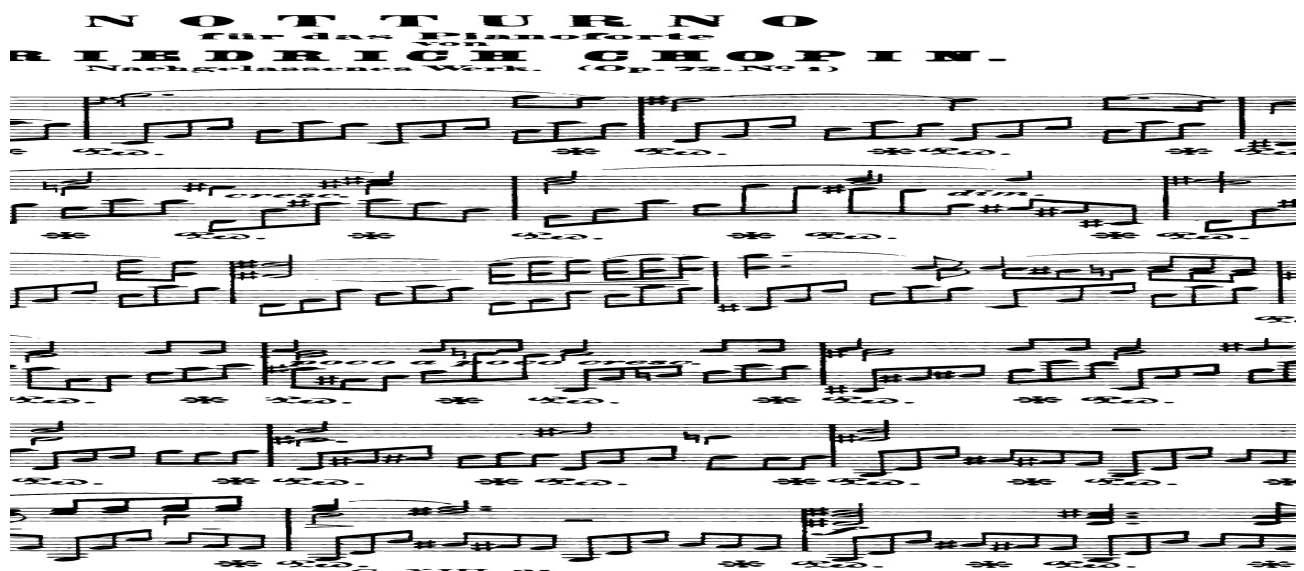
![exemples/plus_haute.png](height:"30%", legend:"Trop haute avec height à 30%")



*Trop haute
avec height à
30%*

EXEMPLE 6 • Une image PNG trop haute, dont on réduit explicitement la hauteur mais qui doit faire la largeur de la page. Elle va bien entendu être complètement déformée...

![exemples/plus_haute.png](height:"30%", width:"100%", legend:"Trop haute avec height à 30% et width à 100%")



Trop haute avec height à 30% et width à 100%

EXEMPLE 7 • Une image plus petite que la largeur sera laissée telle quelle.

![exemples/moins_large.jpg](legend:"Image centrée")



Image centrée

EXEMPLE 8 • Image plus petite en largeur, alignée à gauche et décalée vers la droite, sans légende.

`![exemples/moins_large.jpg](left: "2cm")`



EXEMPLE 9 • Image plus petite en largeur, avec légende, alignée à gauche et décalée vers la droite. Noter comment la légende a été découpée pour passer à la ligne. Noter également l'utilisation d'un ajustement vertical pour l'image (`vadjust`) et pour la légende (`vadjust_legend`).

`![exemples/moins_large.jpg](left: "2cm", vadjust: 1, legend: "Image à gauche décalée
vers
la droite", vadjust_legend: 10)`



*Image à gauche
décalée
vers
la droite*

EXEMPLE 10 • Image plus petite en largeur alignée à droite.

`![exemples/moins_large.jpg](align: :right, legend: "Alignée à droite")`



Alignée à droite

EXEMPLE 11 • Image plus petite en largeur alignée à droite et décalée vers la gauche.

![exemples/moins_large.jpg](right: 2.cm, legend:"À 2 cm de la droite")



À 2 cm de la droite

EXEMPLE 12 • Image plus petite que la largeur, agrandie avec « width: "100%" » pour tenir dans la largeur (déconseillé car cela « abîme » l'image).

![exemples/moins_large_et_bas.jpg](width:"100%", legend:"Image agrandie à 100 % avec une grande légende qui doit tenir sur plusieurs lignes car une légende, par défaut, couvre la moitié de la page, mais on peut changer ce comportement.")

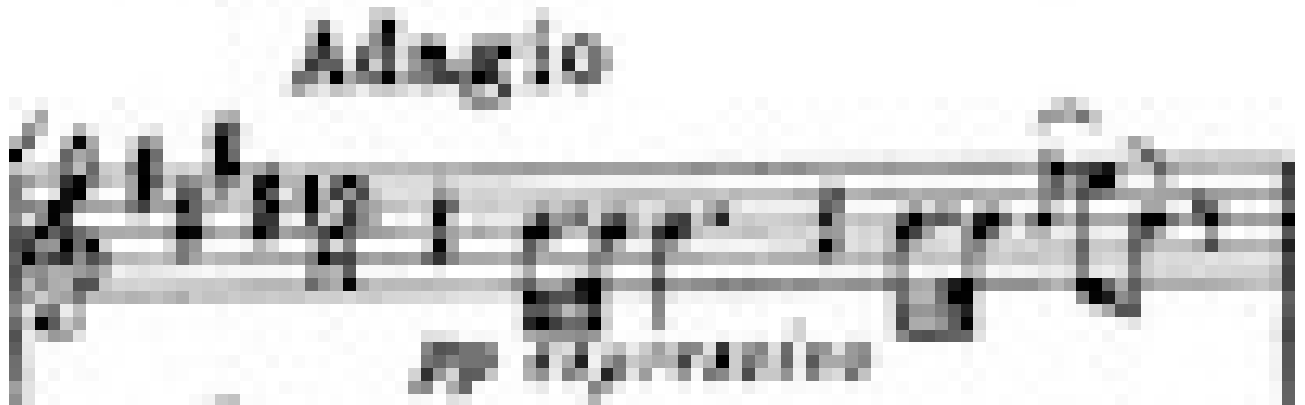


Image agrandie à 100 % avec une grande légende qui doit tenir sur plusieurs lignes car une légende, par défaut, couvre la moitié de la page, mais on peut changer ce comportement.

EXEMPLE 13 • Image plus grande grâce à « scale: 2.5 ».

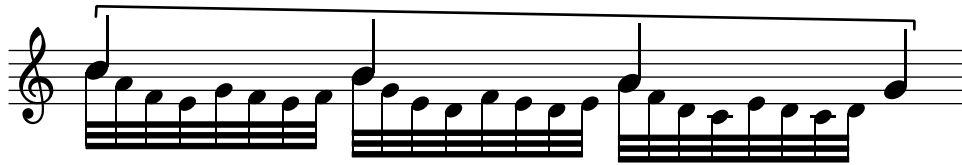
![exemples/moins_large.jpg](scale: 2.5, legend:"Image avec scale de 2.5")



Image avec scale de 2.5

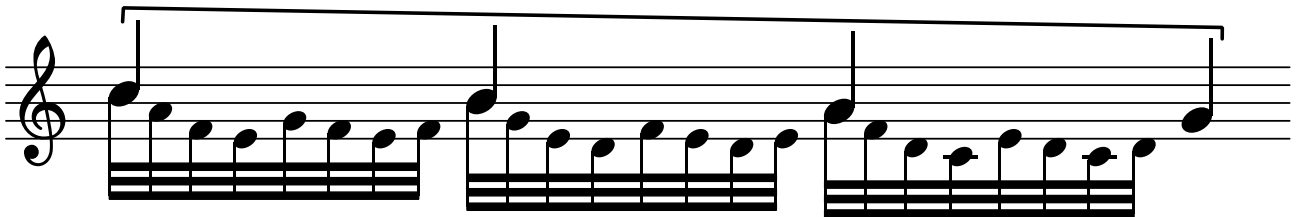
EXEMPLE 14 • Image SVG affichée sans aucune indication.

```
![exemples/image.svg]
```



EXEMPLE 15 • Image SVG grossie.

```
![exemples/image.svg](width: "100%")
```



EXEMPLE 16 • On ne peut pas (à ma connaissance) modifier disproportionnellement une image SVG dans Prawn même avec des valeurs explicites. Ci-dessous, la hauteur, bien que mise à 1000, reste proportionnée à la largeur.

```
![exemples/image.svg](width: 200, height: 1000)
```



EXEMPLE 17 • Image SVG avec légende. Remarquez l'utilisation de `vadjust_legend` (« ajustement vertical de la légende ») qui permet ici de l'éloigner de l'image.

```
![exemples/image.svg](width:"80%", legend:"Image SVG avec  
légende", vadjust_legend: 10)
```

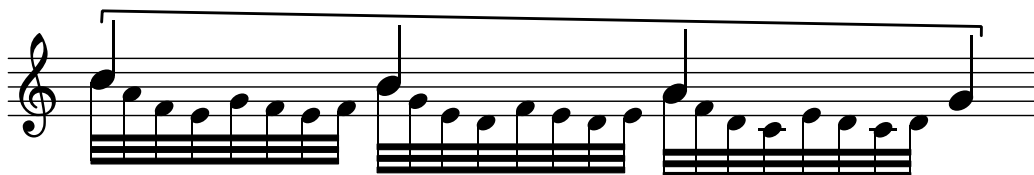


Image SVG avec légende

EXEMPLE 18 • Image avec de l'espace avant et de l'espace après définis par les propriétés `space_before` (« espace avant ») et `space_after` (« espace après »).

```
![exemples/plus_large.jpg](space_before:180, space_after: 100,
legend:"La légende se place toujours bien.")
```

Adagio

The image shows a musical score for a piece titled "Adagio". It consists of five staves. The first staff is in treble clef and contains a melodic line with many beamed sixteenth notes, marked *pp espressivo*. The second staff is in treble clef and contains a line of half notes, marked *pp*. The third staff is in treble clef and contains a line of half notes, marked *pp*. The fourth staff is in bass clef and contains a line of half notes, marked *pp*. The fifth staff is in bass clef and contains a line of eighth notes, marked *pp* and *pizz. sempre*. The score is set in 12/8 time and has a key signature of one sharp (F#).

La légende se place toujours bien.

EXEMPLE 19 • Image sans aucun redimensionnement, grâce à l'option `no_resize`. Cette image « mange » évidemment sur la marge extérieure...

`![exemples/plus_large.jpg](no_resize:true, width:1000, left:0.1)`



Fin des exemples d'images.

Les Images flottantes

Cette partie décrit comment obtenir une image flottante dans le texte. Pour définir une image flottante, il faut définir deux choses indispensables :

- Une image définissant la propriété `float` (cf. ci-dessous),
- Un texte, *avant cette image*, commençant par « ! ».

Les paramètres pour gérer l'image flottante sont les suivants :

- **float** | Définit que l'image est flottante. Ce paramètre peut avoir la valeur `:left` (« gauche » en anglais) pour « flottante à gauche » (l'image sera donc à gauche du texte) ou `:right` (« droite » en anglais) pour « flottante à droite » (l'image sera donc à droite du texte).
- **lines_before** | Définit le nombre de lignes qui vont passer au-dessus de l'image. Par défaut à 0, la première ligne de texte est alignée au haut de l'image (en fonction de ligne de référence du texte).
- **margin_top** | Définit la hauteur à laisser au-dessus de l'image, qu'il y ait ou non des lignes au-dessus. En ne jouant que sur ce paramètre, on ne peut pas faire passer de lignes au-dessus de l'image (utiliser `lines_before` pour ça).
- **vadjust** | Lorsque le `margin_top` ne permet pas de positionner l'image verticalement, on peut se servir de cette propriété *traditionnelle*.
- **margin_left** | Marge à gauche. Quand l'image est flottante à gauche, détermine la distance de l'image avec la marge gauche. Quand l'image est flottante à droite, détermine la distance de l'image avec le texte à gauche. Cette valeur peut être déterminée par défaut dans la recette comme indiqué ci-dessous.
- **margin_right** | Marge à droite. Quand l'image est flottante à gauche, détermine la distance de l'image avec le texte à droite. Quand l'image est flottante à droite, déterminer la distance entre l'image et la marge droite. Cette valeur peut être déterminée par défaut dans la recette comme indiqué ci-dessous.
- **margin_bottom** | Marge en bas entre le bas de l'image et la première ligne du texte associé à l'image. Attention : il s'agit bien du *texte associé à l'image* et non pas d'un paragraphe suivant qu'il faut éloigner de l'image soit avec `space_after` soit avec des `((line))`.
- **text_width** | Largeur que doit occuper le texte à côté de l'image. Si non défini, tout l'espace possible.

Note : on ne permet pas l'utilisation d'une image flottante au milieu, ce qui se fait rarement dans un livre qui n'est pas purement graphique.

POSITIONNEMENT EXACT DE L'IMAGE

On peut déterminer le positionnement exact de l'image de façon générale, dans la recette (pour que toutes les images disposent de la même présentation) ou pour chaque image en particulier.

PASSAGE NATUREL À LA PAGE SUIVANTE

Pour le moment, **Prawn-For-Book** ne gère pas le passage naturel à la page suivante lorsque l'image et son texte enroulé ne tiennent pas dans la page. Il faut donc le gérer *manuellement*, à l'aide de `((line))` et de `((new_page))` par exemple.

ROTATION DE L'IMAGE

Pour le moment, en mode simple, **Prawn-For-Book** ne permet pas d'effectuer une rotation de l'image. Mais cette fonctionnalité sera implémentée plus tard.

Pour le moment, la solution consiste à faire une image déjà tournée et à jouer sur des valeurs négatives pour bien placer le texte. Ou, en mode expert, à l'imprimer explicitement avec du code ruby.

Définition des valeurs par défaut dans la recette

```
---
book_format:
  text:
    # Distance avec l'image flottant à gauche
    left_margin_with_floating_image: 10
    # Distance avec l'image flottant à droite
    right_margin_with_floating_image: 10
```

Exemples divers

Texte enroulé autour de l'image

Un texte avec deux lignes au-dessus de l'image et qui va s'enrouler ensuite par la droite jusqu'au milieu de l'image ensuite. Il est obtenu avec un *float* à *:left*, un *lines_before* à 2 et ensuite toutes les marges à 0. Le *margin_top* à 0 fait que l'image « colle » aux deux lignes supérieures. Le *margin_left* fait que l'image « colle » à la marge gauche et le *margin_right* fait que le texte « colle » à l'image. Et enfin, le *margin_bottom* fait que le texte qui passe sous l'image est collé à l'image comme le texte au-dessus. Noter cependant que le texte, se plaçant toujours sur des lignes de référence, se trouve plus éloigné de l'image en bas car la ligne de référence se trouve plus loin. En fonction du texte, on joue sur les propriétés *margin_top* et *margin_bottom* pour obtenir la meilleure harmonie. Il faut toujours s'arranger pour que l'air au-dessus et au-dessous de l'image soit toujours le même. C'est ce que l'on fait avec l'image suivante, en partant sur les mêmes bases que cette image-ci.



Même chose que ci-dessus, mais la position de l'image a été ajustée avec un *margin_top*: 4 pour être bien au milieu verticalement. Obtenu avec un *float*: *:left*, un *lines_before*: 2 et les deux marges *margin_left* et *margin_right* à 0. Le *margin_left* fait que l'image « colle » à la marge gauche et le *margin_right* fait que le texte « colle » à l'image. Et enfin, le *margin_bottom* fait que le texte qui passe sous l'image est collé à l'image comme le texte au-dessus. Noter cependant que le texte, se plaçant toujours sur des lignes de référence, se trouve plus éloigné de l'image en bas car la ligne de référence se trouve plus loin. En fonction du texte, on joue sur les propriétés *margin_top* et *margin_bottom* pour obtenir la meilleure harmonie. Il faut toujours s'arranger pour que l'air au-dessus et au-dessous de l'image soit toujours le même. C'est ce que l'on fait avec cette image, en partant sur les mêmes bases que l'image précédente.



Même chose que ci-dessus, mais avec les valeurs de marge droite et gauche (*margin_left* et *margin_right*) laissées à leur valeur par défaut, c'est-à-dire respectivement 0 et 10. Donc *:float*: *:left*, position de l'image a été ajustée avec un *margin_top*: 4 pour être bien au milieu verticalement. *lines_before*: 2 pour laisser deux lignes au-dessus, *margin_left* à 0 et *margin_right* à 10. Et enfin, le *margin_bottom* fait que le texte qui passe sous l'image est collé à l'image comme le texte au-dessus. Pour rappel, ce texte s'enroule autour de l'image parce qu'il est précédé d'un point d'interrogation (!Même chose que ci-dessus, [etc.]). En fonction du texte, on joue sur les propriétés *margin_top* et *margin_bottom* pour obtenir la meilleure harmonie. Il faut toujours s'arranger pour que l'air au-dessus et au-dessous de l'image soit toujours le même.



Ce texte commence avec deux lignes au-dessus de l'image car `lines_before` est à 2, puis s'enroule à droite de l'image pour repasser ensuite ses dernières lignes sous l'image.



Aucune marge n'est ajoutée à gauche (`margin_left: 0`) et un espace de 40 (donc plus grand que l'espace par défaut qui est de 10) est défini entre l'image et le texte grâce à la propriété `margin_right`).

De la même manière que pour le haut, on a mis un `margin_bottom` à une grosse valeur pour montrer comment on peut avoir beaucoup d'espace entre le texte et l'image en jouant sur toutes ces propriétés.

On voit aussi grâce à cet exemple comment plusieurs paragraphes peuvent être « enroulés » autour d'une image simplement en les précédant d'un point d'exclamation (voir le code du texte et le rendu). Noter que le nombre de lignes sous l'image n'a pas besoin d'être précisé, contrairement au nombre de lignes au-dessus, puisque c'est simplement tout le texte restant qui sera écrit sous l'image, au besoin. Ici, on n'aura que deux lignes, mais ça pourra dépendre encore des changements de taille du présent manuel.

Texte à côté de l'image



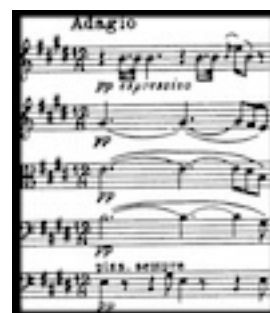
Un texte qui va simplement se placer à droite de l'image, sans dépasser ni en haut ni en bas, avec `float: :left` et `width: 100`. Code:

```
!Un texte qui va simplement [etc.].  
![exemples/image.jpg](float: :left, width: 100)
```

Le paragraphe qui suit l'image se place bien en dessous et, normalement, ne doit pas être mangé par le bas de l'image.

Texte placé à gauche d'une image qui porte le `float` à `:right` (flottante à droite) et le `width` à 100. Code :

```
!Text placé à gauche d'une [etc.]  
![exemples/image.jpg](float: :right, width: 100)
```



Le paragraphe qui suit l'image se place bien en dessous et, normalement, ne doit pas être mangé par le bas de l'image.

Image flottante avec légende

Un texte qui s'enroule autour d'une image flottante à droite qui possède une légende normale et toutes les valeurs par défaut.

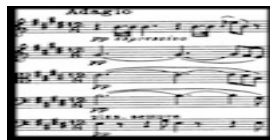
Code utilisé :

```
!Un texte qui s'enroule autour d'une [etc.]
![exemples/image.jpg](float: :right, legend: « La
légende de l'image »)
```



La légende de
l'image

Dans cette exemple d'image avec une légende, le texte s'enroule vraiment autour de l'image c'est-à-dire qu'un `lines_before` à 2 permet de laisser passer deux lignes au-dessus et la longueur du texte fait passer des lignes en dessous de la légende. Un `margin_left` à 20 permet de décoller l'image de la marge gauche tandis qu'un `margin_right` à 20 éloigne un peu le texte horizontalement. Un `margin_bottom` à 8 fait descendre d'une ligne les lignes sous l'image et pour bien positionner verticalement l'image et sa légende, un `vadjust` à 12 fait descendre l'image (ce qui est nécessaire puisque le texte se place toujours sur les lignes de référence — il est donc nécessaire, en fonction de l'image, d'ajuster la position verticale comme on le ferait à la main).



La légende de
l'image

Pour obtenir l'effet ci-dessus, on a utilisé le code :

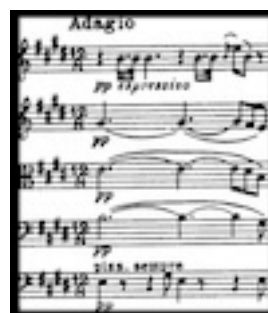
```
!Dans cette exemple d'image avec une légende, le texte [etc.]
![exemples/image.jpg](float: :left, width: 100, height: 50,
legend: « La légende de l'image », lines_before: 2, vadjust: 12,
margin_bottom: 8, margin_left: 20, margin_right: 20)
```

Autre cas d'utilisation



Pour une image qui flotte à gauche (`float: :left`), éloignement de la marge gauche avec un `margin_left` à 40. Le texte est éloigné de l'image avec un `margin_right` à 20 au lieu de la valeur par défaut 10.

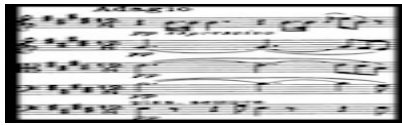
Pour une image qui flotte à droite (`float: :right`), éloignement de la marge droite avec un `margin_right` à 40. Le texte est éloigné de l'image avec un `margin_left` à 20 au lieu de la valeur par défaut 10.



On peut définir la largeur que devra prendre le texte avec la propriété `text_width` qui est mise ici à 200 pour produire un effet de légende sur le côté. Le `margin_left` est à 40, ainsi que le `margin_right`.

Adaptations naturelles du texte

Montrons quelques cas d'adaptation du texte à l'image en fonction de quelques propriétés comme les marges et la largeur du texte.

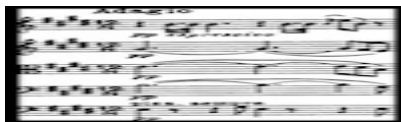


Un largeur de texte normale, un simple `float: left` avec dimensionnement de l'image (pour avoir moins de lignes)

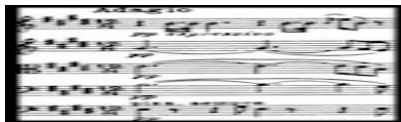
On place une ligne au-dessus (`lines_before: 1`) et une largeur de texte plus réduite (`text_width: 200`) mais qui ne sera donc appliquée que lorsque le texte passera à côté de l'image. Lorsqu'il repassera en dessous, il trouvera sa largeur normale. On a ajouté un petit espace sous l'image (`margin_bottom: 8`) pour que le texte en soit pas trop collé à l'image en dessous (pour qu'il passe sur la ligne de référence suivante).



La même chose que précédemment, mais en éloignant l'image de la marge avec `margin_left: 30` et en éloignant le texte de l'image avec `margin_right: 45`. Toutes les autres valeurs restent similaires et notamment le `margin_bottom: 8` qui permet de passer cette ligne sur la bonne ligne de référence.



Et enfin, toujours en s'inspirant de l'image ci-dessus, on se sert de la propriété `vadjust: 5` (ajustement vertical) pour équilibrer l'espace vertical entre le texte et l'image, pour qu'il y ait le même espace au-dessus et au-dessous et qu'ainsi l'image soit affichée de façon plus équilibrée.



Cas spéciaux

Un texte qui va être placé plus bas, presque au milieu de l'image, grâce à un `margin_top` négatif.



Pour obtenir l'effet ci-dessus, nous avons eu recours à :

```
(( line ))
(( line ))
(( line ))
```

```
(( line ))
!Un texte qui va être placé plus bas, presque au milieu de
l'image, grâce à un `margin_top` négatif.
![exemples/moins_large_border.jpg](float: :right, margin_top: -40)
```



Un texte qui va « manger » sur l'image en mettant un
`margin_right` négatif.

L'effet ci-dessus est obtenu à partir du code :

```
!**<font size="16"><color rgb="FF0000">Un texte qui va « manger »
sur [...] négatif.</color></font>**
![exemples/image.jpg](float: :left, margin_right: -40)
```

Format de la légende

Comme nous l'avons mentionné, par défaut, la légende d'une image se formate avec la police par défaut, la taille diminuée de une unité, le style *italique* et la couleur noire. Mais cet aspect naturel, qui passera pour tous les livres, peut être modifié pour tout le livre, dans la page 19, ou de façon ponctuelle² grâce aux données avant le paragraphe définissant l'image.

² Ce genre de modification doit être absolument évitée pour garder une cohérence dans l'aspect général du livre.

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

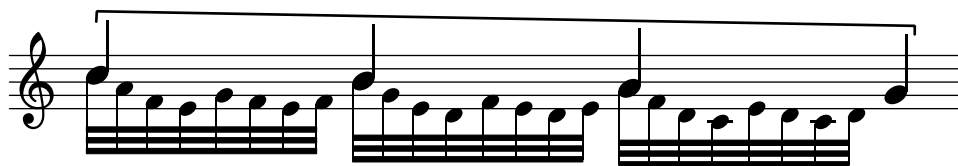
```
---
book_format:
  images:
    legend:
      font: Courier
      size: 8
      color: "0FF0F0"
```

Codes suivis de leur interprétation

```
![exemples/image.svg](legend: "Une légende dans le style défini en recette")
```

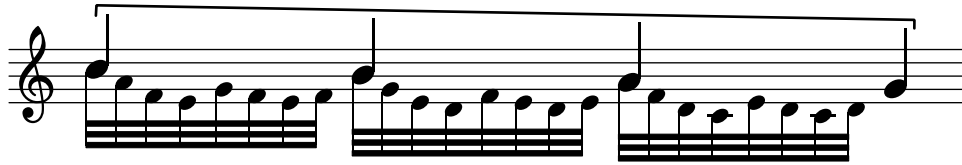


```
![exemples/image.svg](legend: "Légende avec couleur et taille  
ponctuelle", legend_color: [0,110,20,23], legend_size: 22)
```



*Légende avec couleur et
taille ponctuelle*

```
![exemples/image.svg](legend: "Autre fonte, style, taille et  
couleur", legend_font: "Helvetica", legend_size: 14, legend_style:  
:regular, legend_color:[10,10,100,10], vadjust_legend: 15,  
space_before: 10)
```



Autre fonte, style, taille et couleur

Enim excepteur non ea officia sunt nostrud exercitation occaecat aliqua nostrud.

Bibliographies

Introduction

Prawn-For-Book possède un système de gestion des bibliographies très puissant. Elle peut en gérer autant que l'on veut, donc une infinité, et les mettre en forme de la façon exacte que l'on veut, avec une connaissance minimum du langage ruby (et encore, comme d'habitude, on peut se contenter des comportements par défaut).

Une bibliographie se définit dans **Prawn-For-Book** par une donnée dans la section principale de recette `bibliographies` et une balise qui servira d'identifiant pour la bibliographie et de balise pour repérer le texte. Par exemple les balises `article` et `film` qui serviront d'illustration ci-dessous et seront utilisés dans le texte de cette manière :

« C'est une référence à l'article `article(Les chaussures|0001)` et une référence au film `film(Titanic)`. »

Nomenclature pour les bibliographies

(comme pour les autres parties, commençons par un lexique qui permet de fixer les termes que nous employons pour parler de bibliographie)

- **Bibliographie** | Ensemble de sources quelconques, de toutes natures, dont il est fait référence au cours du livre, et qu'ils faut rassembler à la fin du livre, avec ses informations, pour que le lecteur puisse s'y référer. Par exemple une liste de livres, de films ou de mots techniques. Désigne aussi l'ensemble de ses *items*.
- **item** ou **item bibliographique** | C'est un élément en particulier de la bibliographie, un film en particulier s'il s'agit d'une liste de films, un livre en particulier s'il s'agit d'une liste de livres, un mot en particulier si c'est un index de mots.
- **Marque de référence** | Marque spéciale, dans le texte, qui permet de consigner un *item bibliographique*. Elle se construit à l'aide de l'identifiant de la bibliographie, suivi de parenthèses à l'intérieur desquelles on place l'identifiant de l'*item bibliographique*. Par exemple `livre(Les Misérables)` dans la phrase Avez-vous lu `livre(Les Misérables)`, un livre de `person(Victor Hugo)`.
- **Données bibliographiques** ou **Banque de données** | Ensemble complet des *fiches* qui comprennent l'intégralité des *items* d'une bibliographie, même ceux non cités, dans laquelle on va puiser les sources pour y faire référence.
- **Fiche** ou **carte** | Fichier informatique ou enregistrement (dans un fichier CSV) qui contient toutes les données de l'*item bibliographique* (par exemple toutes les données du film ou du livre).
- **Liste des sources** | Désigne la liste, à la fin du livre, qui liste l'ensemble des *items* cités (et seulement les items cités) et, dans **Prawn-For-Book**, indique les pages où il y est fait référence.

Données minimales des bibliographies

Une bibliographie est définie par les données minimales et indispensables suivantes (pour la compréhension, on choisit d'illustrer une bibliographie pour des articles de journaux) :

- Le **tag** (ou **identifiant**) | C'est un mot simple en minuscule (donc seulement les lettres de « a » à « z », au singulier (p.e. « article ») qui sert de *clé* pour la bibliographie. C'est ce tag qui sera utilisé au cours du livre pour construire un *marquer de référence* pour faire référence à un *item bibliographique* en particulier (cf. plus bas).
- **title** (« titre » en anglais) | Le titre de la bibliographie, qui sera imprimé avant d'afficher la bibliographie.
- **path** (« chemin d'accès » en anglais) | qui indique le chemin jusqu'à la *banque des données bibliographiques* qui contient, comme son nom l'indique, toutes les données bibliographiques. C'est au mieux un dossier (contenant les *cartes* ou les *fiches* de données, une par *item*) ou un fichier (contenant toutes les données).
- **main_key** (« clé principale » en anglais) | Si ce n'est pas la propriété `title` de l'*item* (à ne pas confondre avec le `title` dont nous venons de parler) qui doit être utilisée pour remplacer la marque de référence, c'est la valeur de cette clé principale qu'il faudra prendre (et qui doit donc impérativement être définie pour chaque *item bibliographique*)

Ce sont les données minimales à définir pour qu'une bibliographie soit utilisable. Dans la recette (cf. page 19), elles sont définies de cette manière :

bibliographies:

```
<tag>:
# Propriétés indispensables
title: "<titre>"
path: "<path/to/data>"
main_key: « <clé item> » # si autre que title
# Propriétés optionnelles
format: <format>
picto: "<picto>"
```

À partir de là, dans le texte, il suffit de mettre un mot ou un grand de mot entre parenthèses en le précédant de l'identifiant de la bibliographie pour que cet élément soit pris en repère. Par exemple :

```
<id biblio>(<id item>)
```

Ce mot sera remplacé par la propriété `title` de l'*item* (cf. plus bas) et cette référence sera enregistrée pour l'ajouter à la liste des sources bibliographiques à la fin du livre.

Si le `title` de l'*item* ne convient pas dans une utilisation particulière, on peut définir un texte quelconque avant l'identifiant, en le séparant avec un trait droit (« | ») de l'identifiant, comme ci-dessous.

```
C'est un film(exemple de film|the titanic) dans le texte.
```

Avec le code ci-dessus, le texte « exemple de film » sera utilisé à la place du titre du film, mais c'est une référence au film en question qui sera enregistrée en liste des sources en fin de volume.

DONNÉES OPTIONNELLES

Pour commencer à personnaliser l’affichage de la bibliographie (ou des *items* dans le texte), on peut utiliser ces propriétés optionnelles :

- **picto** | Pictogramme à utiliser avant le texte qui remplacera la *marque de référence* dans le texte du livre.
- **title_level** (« niveau de titre » en anglais) | Le niveau de titre pour la section qui liste les sources citées. Par défaut, ce niveau est 1, le plus grand titre.
- **format** | Format à utiliser pour l’affichage de la *liste des sources*,
- **key_sort** (« clé de classement » en anglais) | Clé de classement des *items* dans la *liste des sources*

DONNÉES BIBLIOGRAPHIQUES

Dans l’idéal, les données sont consignées dans un dossier qui contient chaque donnée sous forme de fiche (un fiche par item) au forme YAML (le format des recettes). Ce format permet une édition facile des données. Cependant, vous pouvez aussi utiliser le format JSON ou même TXT (simple texte).

ITEMS DE LA BIBLIOGRAPHIE

Au minimum (mais ce serait un peu idiot de n’avoir que ça...), un item de bibliographie doit définir sa propriété `title` qui correspond à son titre. `title` est sa clé principale. Mais puisque **Prawn-For-Book** est hautement configurable, même cette clé principale peut avoir un autre nom, il suffit de la définir dans la propriété `main_key` des données de la bibliographie, dans la recette.

Donc un item bibliographique peut ressembler à :

```
# Dans "the titanic.yaml"
---
title: "The Titanic"
title_fr: "Titanic"
director: "James CAMERON"
year: 1999
```

LISTE DES SOURCES

Dans le livre, il suffit de placer le code `((id bibliographie>))` pour insérer la bibliographie à l’endroit voulu. Par défaut, toutes les informations seront affichées, mais il sera possible de tout formater à sa convenance (cf. plus loin).

Par exemple :

```
(( biblio(film) ))
```

... pour afficher la liste des films cités (et seulement ceux cités) et/ou :

```
(( biblio(article) ))
```

... pour afficher la liste des articles (seulement ceux cités), avec les pages où ils sont cités.

FORMATAGE DE LA *LISTE DES SOURCES*

Le formatage de l’affichage de la *liste des sources* en fin d’ouvrage se définit, comme nous l’avons vu, par la propriété `format` de la bibliographie, dans la recette.

Cette propriété n’est pas obligatoire et en son absence, ne sera affiché à la fin du livre que le titre de l’*item* (propriété `title`) ainsi que la liste des pages (ou autre indication en fonction de la `_pagination_` choisie) où cet *item bibliographique* est cité. C’est le cas, dans l’exemple, pour la liste des articles par exemple (cf. ci-dessous).

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

```
---
bibliographies:
  # Ici la définition de toutes les bibliographies

  # Définition de la première bibliographie

  article: # id bibliographie
    title: "Articles de presse"
    title_level: 2
    path: "chemin/vers/données/articles"

  # Définition de la deuxième bibliographie

  film:
    title: "Liste des films"
    path: "chemin/vers/données/films"

  # Définition d’une autre bibliographie

  autrebib:
    title: "Pour propose autre chose"
    path: "..."
    picto: :fiche
    main_key: "truc" # autre clé que 'title'
```

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

Un texte exemple qui utilise un article(premier article) en référence bibliographique. Ce paragraphe contient aussi une référence au film film(The Titanic) qui se déroule sur un bateau.

Le livre final (document PDF) contiendra :

Un texte exemple qui utilise un Mon Premier Article en référence bibliographique. Ce paragraphe contient aussi une référence au film The Titanic qui se déroule sur un bateau.

Personnalisation des bibliographiques

Comme pour tout élément de **Prawn-For-Book**, on peut le garder simple ou au contraire le formater tel qu'on le désire. C'est pratiquement incontournable pour les bibliographies, au moins au niveau de la *liste des sources*.

DÉFINIR LA FONTE

Il n'est pas très heureux de modifier la fonte à l'intérieur d'un texte, ça le rend beaucoup plus compliqué à lire et même à afficher. En revanche, on peut profitablement introduire des pictogrammes discrets qui permettent de caractériser les éléments.

FORMATAGE DE LA LISTE DES SOURCES

Si la propriété `format` de la bibliographie est fournie dans la recette, elle se sert de marques de type `%{<propriété>}` pour définir le format à utiliser, où `<propriété>` est le nom de la propriété dans les données de l'*item bibliographique* (à commencer par `title`, son titre).

Lorsque la valeur doit être modifiée, par exemple mise en capitales, on ajoute la méthode de transformation après un trait droit « | » :

`%{propriété|méthode transformation}`
(voir ci-dessous les méthodes existantes).

Par exemple :

```
format: "%{title|all_caps} (%{year} - %{year|age}, réalisation :
%{director|person}, acteurs : %{actors|person}, durée :
%{duration|horloge})"
```

Ce *format* va afficher le titre, tout en majuscules (`all_caps`), puis, entre parenthèses, l'année de sortie (`year` qui signifie « année » en anglais), le réalisateur (précédé du terme « réalisation : » et traité comme un individu) et enfin la liste des acteurs et actrices, traités comme des individus, précédée de la marque « acteurs : ».

Optionnellement, si la liste des références ne doit pas s'afficher à la suite de l'*item* après un double-point, ajouter `%{pages}` à l'endroit voulu.

Vous pouvez voir ci-dessous le rendu de ce formatage.

MÉTHODE DE FORMATAGE COMMUNES

- `all_caps` | Met la donnée en majuscule.
- `person` | Transforme une donnée « Prénom NOM » en « Prénom Nom » ou une liste de personnes de la même manière.
- `age` | Transforme une donnée « année » en âge par rapport à maintenant.
- `horloge` | Transforme une donnée secondes en horloge de type `h:mm:ss`.
- `minute_to_horloge` | Transforme une donnée minutes en horloge de type `h:mm:ss`.

En mode expert (cf. page 86) vous pouvez également définir toutes formes de méthodes de transformation.

Extrait de la recette

```
# ...
bibliographies:
  article:
    title: Articles de presse
    title_level: 2
    path: biblios/articles
    picto: :fiche
  film:
    title: Films cités
    title_level: 2
    path: biblios/films
    picto: clap
    format:      "%{title|all_caps}      (%{year}      – %{year|age},
réalisation :  %{director|person},
    acteurs :      %{actors|person},      durée :
%{duration|minute_to_horloge}). Occurrences :
    %{pages}"
  costum:
    title: Liste des costumes
    title_level: 3
    path: biblios/costums
    main_key: nom
    format: "%{nom|transforme_nom}"
```

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

```
Un costum(cravate) pour voir.
(( new_page ))
(( biblio(costum) ))
(( new_page ))
```

Le livre final (document PDF) contiendra :

Un Cravate pour voir.

Liste des costumes

Cravate : 81.

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 `PFLError.context = "Le contexte"`.

*Exemple dans un helper***

```
# ruby
def monHelper(pdf, book)
  12.times do |i|
    # Indiquer le contexte
    PFLError.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
  PFLError.context = nil
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 page¹.
-

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.

Évaluation au second tour

Certaines opérations ou certains calculs ne peuvent s'opérer qu'au second tour de l'application³ — typiquement, le nombre de pages du livre —. On utilise alors la tournure suivante pour réaliser ces opérations.

```
#{{{ "#{operation}" if Prawn4book.second_turn? }}}4
```

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.

³ 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.

⁴ Noter ici l'utilisation des trois accolades obligatoires lorsque le code lui-même a recours aux accolades.

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

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')}`.

Le livre final (document PDF) contiendra :

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 11 12 2023.

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`

Tutoriel

Cette section présente un tutoriel de prise en main de l'application **Prawn-For-Book** qu'il suffit de suivre pour réaliser confortablement son premier livre professionnel.

Pré-requis du tutoriel

Pour pouvoir réaliser ce tutoriel, vous n'avez besoin de rien d'autre que l'application **Prawn-For-Book** elle-même, correctement installée.

Vous devez également avoir des rudiments concernant l'utilisation du Terminal (sur MacOS) ou de la Console (sur Windows), mais nul besoin d'être un expert pour produire un bon livre !

Pour vous assurer que **Prawn-For-Book** est bien installé, ouvrez simplement une fenêtre de Terminal (ou une Console) et tapez :

```
> pfb -version
```

(remarquez que tous les codes qui seront précédés de « > » seront à exécuter dans votre console de terminal)

Cette première commande **Prawn-For-Book** devrait vous afficher la version de l'application que vous utilisez. Si cette version n'est pas « 2.0.0 », vous devriez l'actualiser pour être à jour des dernières fonctionnalités.

Si ce n'est pas le cas, alors il faut que vous recommenciez l'installation en suivant la procédure proposée en annexe (cf. ### REF: annexe_installation_application ###).

Si tout est OK, vous pouvez vous lancer à corps et à cri dans ce tutoriel !

Bon tutoriel et bonne découverte !

Premiers pas

Dans ce tutoriel, nous allons concevoir notre premier livre publiable qui contiendra tout ce qu'il faut savoir sur **Prawn-For-Book** pour pouvoir démarrer et se débrouiller. À l'issue de ce tutoriel, vous serez en possession d'un fichier PDF que vous pourrez publier facilement. Si vous êtes inscrit au programme KDP d'Amazon (cf. ### REF: annexe_kdp_amazon ###), vous pourrez même aussitôt en demander des épreuves papier.

Dans cette première partie, nous allons installer la base de notre livre. Cette base consiste à créer un nouveau dossier sur votre ordinateur (*les opérations seront détaillées ci-dessous, pour le moment, vous pouvez ne faire que lire*) dans lequel nous créerons les deux fichiers de base de l'application, le fichier `texte.pfb.md` pour mettre le texte et le fichier `recipe.yaml` pour définir la recette de ce livre.

À VOUS DE JOUER !

- Ouvrez une fenêtre de Terminal (sur Mac/Unix) ou une Console (sur Windows) dans le dossier dans lequel vous voulez créer votre livre, par exemple le dossier « Documents » (*sur MacOS, control-cliquez sur le dossier Documents dans le Finder et choisissez l'item du menu contextuel qui s'appelle quelque chose comme "Nouveau terminal au dossier"*).
- Que vous soyez sur MacOS, Windows ou Unix, pour simplifier, nous appellerons toujours cette fenêtre la *console*.

- Tapez dans cette console la commande :
`> pfb init`
 ... qui permet d'initier un nouveau livre dans le dossier courant.
- À titre indicatif, **pfb** sont simplement les trois premières lettres de **Prawn-For-Book**.
- L'application vous demande de définir le nom du dossier de votre livre. Vous pouvez lui donner le nom du livre ou tout autre valeur significative. Nous l'appellerons « Nouveau livre ».
- PFB vous demande de confirmer le chemin d'accès, il suffit de presser la touche Entrée sans rien écrire. Si vous n'êtes pas d'accord avec le lieu, il suffit de taper « n » et de recommencer.
- PFB vous présente alors une liste d'opérations avec le titre « DONNÉES DE LA RECETTE ». C'est un assistant qui vous aide dans la conception de votre livre.
- Nous n'allons pas utiliser d'assistant, aussi vous pouvez choisir l'item « Finir » en cliquant tout de suite sur la touche Entrée.
- Comme vous pouvez le constater, PFB construit deux fichiers, l'un pour le texte, l'autre pour la recette, comme nous en avons parlé.
- PFB vous demande ensuite le titre du livre, vous pouvez entrer « Mon premier livre » ou tout autre titre, puis presser la touche Entrée.
- PFB finalise alors le dossier en créant quelques fichiers supplémentaires.
- Elle vous présente ensuite une aide concernant les commandes essentielles. Vous pourrez afficher une liste plus complète dès que vous en avez besoin avec la commande « `> pfb aide` » (*essayez justement cette commande dès à présent*).

BRAVO ! Vous venez de créer votre premier dossier de livre **Prawn-For-Book** !

Nous devons maintenant *rejoindre* notre nouveau dossier pour y travailler notre livre. Soit vous ouvrez une nouvelle console dans le dossier du livre comme nous l'avons fait tout à l'heure, soit, dans la console actuelle, vous tapez...

```
> cd « Nouveau livre »
```

... pour rejoindre ce dossier (*dans le cas où vous l'ignoreriez, "cd" est une commande générale qui n'est pas propre à PFB et qui permet de rejoindre un dossier sur votre ordinateur*).

Revenons à votre bureau (votre *Finder* sur MacOS) pour voir ce que contient votre dossier livre.

Il contient plusieurs fichiers qui nous permettront en temps voulu de formater le livre et d'effectuer certaines opérations très pratiques.

Pour le moment, nous devons seulement nous concentrer sur les fichiers « `texte.pfb.md` » et « `recipe.yaml` ». Ce sont les deux fichiers que nous allons commencé à éditer.

Attention à l'édition !

Pour modifier les fichiers de PFB, vous allez utiliser un *IDE*, c'est-à-dire un *environnement de développement intégré*, c'est-à-dire, pour faire simple, un éditeur qui ne va rien ajouter en douce à votre fichier, qui va le laisser tel que vous l'avez écrit, sans code caché.

En contrepartie, cet IDE, si vous désirez l'exploiter, vous proposera des outils très intéressants, qui vous deviendront bientôt indispensables, pour rédiger votre livre et le mettre en forme comme vous le voulez. Par exemple, il utilisera la *coloration syntaxique* qui vous permet d'y voir très clair dans vos fichiers, en mettant les différents éléments en couleur (pour PFB par exemple, vous pourrez voir le texte du livre proprement dit en noir et les autres éléments dans une autre couleur).

Nous utilisons pour ce faire l'application « Sublime Text » mais vous pouvez utiliser n'importe quel autre IDE. Ils sont très simples d'accès et d'utilisation : vous ouvrez votre

fichier dans cet IDE (par le menu « Ouvrir », ou en glissant le fichier sur l'application), vous tapez votre texte, ou votre code, vous enregistrez, et c'est tout.

Nom de l'auteur et début de texte

Nous allons commencer justement à éditer nos deux fichiers principaux pour définir l'auteur(e) du livre et commencer à lui coller du contenu.

- Ouvrez le fichier « `recipe.yaml` » dans votre IDE (que nous appellerons simplement *éditeur* par la suite).

Astuce : plutôt que d'avoir à ouvrir vos fichiers un par un pour pouvoir les éditer, le plus simple est d'ouvrir tout le dossier du livre dans l'IDE, et de choisir, toujours dans l'IDE, le fichier à voir et modifier.

- Donnez votre nom à la propriété « `author` ("auteur") de la section `book_data` » (« données du livre ») de votre fichier. Votre fichier devrait ressembler à quelque chose comme :

```
---
:app_name: Prawn-For-Book
:app_version: 2.0.0
:created_at: '2023-12-11'
#book_data> 5
book_data:
  titre: Mon premier livre
  author: "John DOE"
---
#/book_data>
```

⁵ Notez que toutes les lignes de ce fichier qui commencent par « # » sont justes des commentaires pour vous repérer et laisser des notes.

Annexe

Marques markdown

Vous trouverez ci-dessous toutes les marques Markdown utilisables.

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

La table ci-dessous a été construite aussi en markdown, en utilisant le format :

```
| CA1 | CA2 | CA3 | CA4 |  
| CB1 | CB2 | CB3 | CB4 |  
|/|
```

Vous pouvez trouver toutes les informations sur l'utilisation des tables à la page [### REF: tables ###](#).

Le livre final (document PDF) contiendra :

italique	<code>* ... *</code>	<i>texte en italique</i>
gras	<code>** ... **</code>	texte en gras
souligné	<code>~ ... ~</code>	<u>texte souligné</u>

Le format YAML de la recette

exposants 1^{er} 1^{er}

Le format **YAML** est un format très simple de présentation et de consignation des données. Il est utilisé dans **Prawn-For-Book** pour définir [### REF: recette_juste_titre ###](#), que ce soit pour un livre unique (cf. page page 19) ou pour une collection (cf. page page 19). Les données sont *imbriquées*, comme nous l'avons dit, pour s'y retrouver plus facilement, entendu que les recettes peuvent contenir de nombreuses informations. Voyez l'imbrication donnée en exemple ci-dessous.

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

```
---  
table_de_donnees:  
  sous_ensemble_liste:  
    - premier item  
    - deuxième item  
  sous_ensemble_texte: "Ma donnée texte"  
  un_nombre: 12  
  une_date: "2023-11-22"  
  quelquun:  
    prenom: "Marion"  
    nom: "MICHEL"
```


Les "Fontes-Strings"

Pour définir les polices dans les éléments, à commencer par la recette, on utilise de préférence ce qu'on appelle dans **Prawn-For-Book** les « **fonte-string** » (« string » signifie quelque chose comme « caractère » en anglais).

Ces **fonte-strings** se présentent toujours de la même manière, par une chaîne de caractères (de lettres) contenant 4 valeurs séparées par des balances, dans l'ordre :

- le **nom** de la police,
- le **style** de la police (qui doit être définie,
- la **taille** à appliquer au texte (en points-pdf),
- la **couleur** éventuelle (pour la définition de la couleur, voir page 97).

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

```
---
font: "<police>/<style>/<taille>/<couleur>"
# Par exemple
font: "Numito/bold_italic/23/FF0000"
```

Définition de la couleur

En règle générale, la couleur dans **Prawn-For-Book** peut se définir de deux façons :

- en hexadécimal (comme en HTML) à l'aide de 6 chiffres/lettres hexadécimal (donc de 0 à F),
- en quadrichromie, avec CMJN (Cyan, Magenta, Jaune, Noir).

COULEUR HEXADÉCIMALE

En hexadécimal, on a 3 paires de deux chiffres/lettres qui représentent respectivement les quantités de rouge, vert et bleu. Par exemple « AAD405 » signifiera « AA » de rouge, « D4 » de vert et « 05 » de bleu.

"000000" représente le noir complet, « FFFFFFFF » représente le blanc complet. Lorsque toutes les valeurs sont identiques (par exemple « CCCCCC ») on obtient un gris (mais il existe d'autres moyens d'obtenir du gris).

Quelques couleurs hexadécimales aléatoires :

- Couleur 1a8243
- Couleur ebedb8
- Couleur 209432
- Couleur d32f60
- Couleur af81f1
- Couleur 573bce
- Couleur e4e701
- Couleur 727cc7
- Couleur ea8626
- Couleur 59251f

COULEUR QUADRICHROMIQUE

La quadrichromie, représentée souvent par « CMJN » (ou « CMYK » en anglais), est le format de la couleur en imprimerie. On aura une liste (crochets) contenant les 4 valeurs de 0 à 127 pour le Cyan (C), le Magenta (M), le Jaune (Y) et le noir (K). Par exemple « [0, 12, 45, 124] » signifiera qu'il n'y aura pas de Cyan, qu'il y aura 12 de magenta, 45 de jaune et 124 de noir. [0, 0, 0, 0] représente le blanc complet, [127, 127, 127, 127] le noir complet.

Quelques couleurs quadrichromiques :

- Couleur [0,0,0,127]
- Couleur [127,0,0,127]
- Couleur [127,0,0,0]
- Couleur [0,127,0,127]
- Couleur [0,127,0,0]
- Couleur [0,0,127,127]
- Couleur [0,0,127,0]
- Couleur [127,127,127,127]
- Couleur [127,127,127,0]
- Couleur [67, 105, 79, 45]
- Couleur [28, 94, 7, 94]
- Couleur [56, 14, 87, 112]
- Couleur [124, 31, 21, 50]
- Couleur [86, 107, 60, 1]
- Couleur [73, 98, 118, 86]
- Couleur [36, 82, 37, 73]
- Couleur [114, 9, 122, 88]
- Couleur [92, 7, 16, 2]
- Couleur [87, 22, 76, 14]

Rogner une image SVG

Il est fort possible qu'en produisant une image SVG, et en l'insérant dans le livre, elle laisse voir trop de blanc autour d'elle, comme dans le premier exemple donné ci-après. Pour palier ce problème, il faut « rogner » cette image SVG. Mais *rogner* une image SVG ne se fait pas aussi facilement qu'avec une image d'un format non vectoriel (JPG, PNG, etc.). Il faut pour ce faire utiliser, après avoir chargé la commande `inkscape` dans votre ordinateur, le code suivant :

```
inkscape -l -D -o image-rogned.svg image.svg
```

Le livre final (document PDF) contiendra :

Image non rognée :



(note : la page vide précédente est occupée par le blanc de l'image rognée)

La même image, rognée cette fois :



Amet exercitation ut dolore in in nulla adipisicing laborum.

Synopsis de création

Cette section présente une sorte de synopsis que vous pouvez suivre pour créer votre livre.

- ...
- ajustez parfaitement vos images en jouant sur leur `vadjust` :. N'hésitez pas à ajouter des lignes avec `((line))` pour bien séparer les choses si nécessaire
- ...

Liste exhaustive des fonctionnalités

- définition de la taille du livre
- pagination automatique et personnalisable
- définition de la fonte par défaut
- définition des pages spéciales à insérer page 25
- justification par défaut des textes
- colorisation des textes
- suppression des veuves et des orphelines
- suppression automatique des lignes de voleur
- nombreuses sortes de puces et puces personnalisées page 41
- évaluation à la volée du code ruby (pour des opérations, des constantes, etc.)
- traitement des références croisées
- traitement dynamique des références à d'autres livres
- gestion par défaut ou personnalisée des images et de leur légende
- génération dynamique de contenu
- corrige l'erreur typographique de l'apostrophe droit
- corrige l'erreur typographique de l'absence d'espace avant et après les chevrons
- corrige l'erreur typographique de l'espace avant et après les guillemets droits et courbes
- corrige l'oubli de l'espace avant les ponctuations doubles
- corrige l'erreur d'espace avant les ponctuations doubles (pose d'une insécable)
- corrige l'absence d'espace insécable à l'intérieur des tirets d'exergue
- changement de fonte (police) pour le paragraphe suivant `### REF: change_fonte_for_next_paragraph ###`
- placement sur n'importe quelle ligne de la page
- exportation seulement du texte produit
- exportation comme livre numérique (pur PDF)

Films cités

THE TITANIC (1999 — 24 ans, réalisation : James Cameron, acteurs : Kate Wistlet et Leonardo Dicaprio, durée : 2:56:48). Occurrences : : 79.

Articles de presse

Mon Premier Article : 79.

AVANT-PROPOS	11
FONCTIONNALITÉS	15
Généralités	17
Les Forces de <i>Prawn-For-Book</i>	17
Les deux fichiers de base	17
Les Recettes	19
Recette du livre	19
Recette de la collection	19
Définition des marges	21
Définition de la fonte par défaut	23
Éditeur / Maison d'édition	23
Le Texte du livre	24
Texte au format pseudo markdown	24
Les Pages spéciales	25
Les Types de pages spéciales	25
Table des matières	26
Page des crédits (Colophon)	29
Page de titre	32
Entêtes et Pieds de page	35
Description	35
Exemple complexe	38
Le Texte en détail	41
Les puces	41
Les Images	49
Insérer une image	50
Les Images flottantes	64
Bibliographies	76
Introduction	76
Liste des costumes	82
Le Mode Expert	84
Description	84
Indication du contexte d'erreur	84
Injection	84
Évaluation du code ruby	85

Bibliographies en mode expert	86
TUTORIEL	89
Premiers pas	90
ANNEXE	95
Marques markdown	96
Le format YAML de la recette	96
Les "Fontes-Strings"	97
Définition de la couleur	97
Rogner une image SVG	98
Synopsis de création	101
Liste exhaustive des fonctionnalités	101
Films cités	102
Articles de presse	102

Icare Éditions

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

Mise en page
Prawn-for-book

Couverture
MM & PP

Correction & relecture
Marion Michel

Imprimé par Prawn-For-Book