ADAPTATIVE LAYOUT

Spécifications EPUB
pour la réalisation
de mise en page liquide

Mars 2014

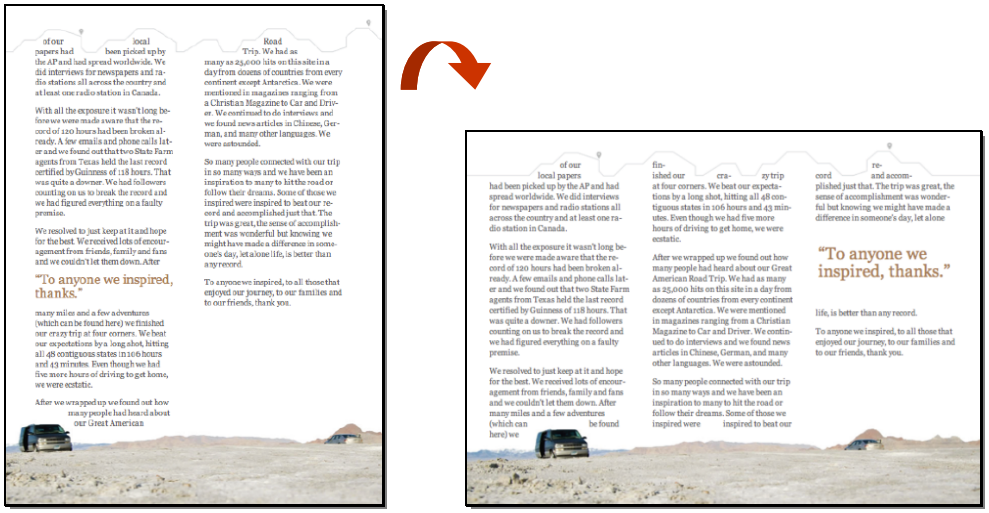
Concept de mise en page liquide ou EPUB Adaptative Layout

d'après EPUB Adaptive Layout, www.idpf.org/epub/pgt

Introduction

Le but du concept de mise en page liquide ou adaptative layout est de permettre un positionnement différent des éléments de la page en fonction de la taille et de l'orientation du Reader, et au delà, une mise en forme différente de ces éléments.

Exemple :



Naturellement, l'idée est de passer par le biais de modèles ou gabarit de mise en page.

Ce concept est utilisé actuellement dans les sites web modernes qui doivent se plier aux différents supports de l'internet mobile. Dans un autre registre, les applications pour tablettes ont elles-aussi abordé le sujet en permettant, selon les cas, un positionnement ou une mise en page différente en fonction de la tablette et de son orientation.

Adobe Indesign intègre depuis la CS6 cette notion de mise en page liquide en permettant à un même contenu d'être récupéré pour une autre mise en page (par exemple paysage ou portrait) sans que le contenu soit différent, ceci par l'intermédiaire de gabarits de pages multiples, mais cela ne concerne pas le format EPUB.

En effet, jusqu'alors le format EPUB ne s'entendait que sous la forme *reflow* ou *fixed-layout* pour son opus 3. Aussi dans les discussions de l'IDPF ce sujet a commencé à poindre avec une orientation CSS.

Aujourd'hui, le sujet n'est pas clos, et il est fort possible que ce concept n'aille pas plus loin dans l'avenir. En outre on peut aussi toujours constater que plusieurs personnes confondent allègrement les 3 concepts de mise en page.

Dans un billet du forum IDPF on peut lire *(http://idpf.org/forum/topic-1301)* :

FIXED LAYOUT AND LIQUID EPUB3

Submitted on October 30, 2013 - 5:28am
#1

Hello, I have seen a PDF document from IDPF (http://idpf.org/sites/default/files/digital-book-conference/presentation...) where I read that EPUB 3 fixed layout is done but EPUB 3 liquid is not.
This document is dated in 2012 so I don't know if there are news about this topic.
So, my main question is about the different kinds of EPUB3, what kinds of EPUB3 exist? - EPUB3 fixed layout - EPUB3 liquid - EPUB3 interactive books
What are the main differences between them?? or their main features?? is there some kind more??
Regards

Submitted by matt.garrish on October 31, 2013 - 8:38am
#2

To try and break them down:

- An interactive ebook typically is boiled down to one that is scripted, but the term itself generally refers to the ability for any form of user interaction with the content -- drag/drop, quizzes, forms, interactable 2d/3d images, etc.
- A fixed layout document is a content document that represents a single page with fixed dimensions, allowing, for example, content to be absolutely positioned on top (i.e., no dynamic pagination or reflowing of the content, no matter the screen size).
- Liquid layouts (or adaptive layouts) refer to page templates, where you can patition up a page and content gets flowed into the regions (e.g., you could have a call-out area in the center and a two-column layout).

An interactive book is just a concept, and as EPUB 3 supports scripting it supports interactive ebooks.
Fixed layouts require actual metadata to create, and EPUB also supports those (and fixed layouts are supported to varying degress in the major reading systems). You can read the FXL spec for more info, but in 3.0.1 this metadata has been integrated into the core specs.
There is also a specification for adaptive layouts, but I don't believe anyone supports them and I'm not sure if or when they will be in the future.

Comme le dit *Matt Garrish*, très actif au sein du consortium et expert pour l'IDPF, il n'est pas certain qu'il existe une application de lecture supportant les mises en page liquides, pire encore, il est encore moins sûr du devenir de cette spécification.

CSS Adaptative Layout

Ce concept de mise en page a fait l'objet d'une spécification *(www.idpf.org/epub/pgt/)* le 08.08.2012 ainsi qu'un brouillon nommé *CSS Page Templates* le 19.01.2012 *(www.idpf.org/epub/pgt/csspgt-20120119.html)*.

La spécification repose sur 2 notions : la mise en page et le style adaptatif.

La mise en page définit comment une page est partitionnée, comment le contenu est associé à une partition spécifique et ajoute des partitions dotées de styles particuliers.

Le style adaptatif va utiliser 2 types de règles stylistiques permettant d'adapter la mise en page (et d'autres éléments de style) en accord avec son environnement (l'écran utilisateur, l'orientation etc.) :

- des règles stylistiques qui s'appliquent en fonction d'un ensemble de conditions
- des règles stylistiques qui vont découler des calculs liés aux paramètres de l'environnement.

1. La mise en page

Cette spécification repose sur plusieurs modules CSS :

- Régions (Regions) et Exclusions (Exclusions)

  CSS Regions et *CSS Exclusions* définissent des zones de texte avec des mises en forme particulières dans les pages. Elles permettent par exemple de justifier finement un texte autour d'une image. Cependant ces modules sont mis à mal par Google qui pense les rejeter car non compatibles avec son nouveau moteur de rendu BLINK *(actualité de developpez.com en date du 28 janvier 2014)*. Google a d'ailleurs reçu le soutien de Håkon Wium Lie, fondateur du CSS et le Directeur Technique d'Opera, qui explique que les *CSS Regions* ne se comportent pas bien avec d'autres aspects du HTML et du CSS. En outre si une mise en page est trop complexe pour les mécanismes de mise en page simples, celui-ci estime qu'il faut la reconsidérer. Ceci expliquerait, entre autres choses, pourquoi Matt Garrish n'est pas si sûr du devenir de l'Adaptative Layout.

- Multi-colonnage (Multi-column Layout)

  Il est quant à lui à peu près bien supporté par l'ensemble des navigateurs et nécessite néanmoins quelques préfixes (-webkit- et -moz-). Un tutoriel de *Alsacréation* est disponible ici *(http://www.alsacreations.com/tuto/lire/1557-les-multicolonnes-en-css3.html)*.
  Cependant ce module n'a de sens dans le cas de la mise en page fixe puisque dans celle-ci la page est déjà prédéterminée. Ainsi les colonnes ne tiendront pas compte du texte de la page suivante si le multicolonnage se poursuit. Le problème survient lors des mises page de type *reflow* où le texte change en fonction de l'affichage et donc les colonnes sont tronquées au lieu de voir les colonnes se remplir de texte de manière adaptée comme on s'y attendrait.

- Transformations 2D (2D Transforms)

  Il s'agit ici d'un module permettant faire réaliser aux éléments des translations, des rotations, des inclinaisons ou des homothéties.

- Fond et bords (Backgrounds and Borders)

  Pas la peine de s'étendre ici sur ces notions de fonds et de bordures déjà amplement connues.

La spécification est aussi en lien avec :

- Media Queries

  La règle du "@-epubx-when" permet une mise en page dynamique en fonction de l'orientation. Elle empiète sur le domaine des *Media Queries* qui possèdent beaucoup plus de propriétés cependant elles ne rentrent pas en conflit. Il faut plutôt voir cette règle comme une propriété supplémentaire.

  D'ailleurs si la propriété @media du CSS est définie de telle sorte qu'elle prévoit une mise en page spécifique en fonction de l'orientation, il n'est pas nécessaire d'utiliser la règle @-epubx-when. Soulignons que les spécifications EPUB 3.0 requièrent le support des *Media Queries*.

  On peut donc se demander si l'utilisation de cette règle est vraiment justifiée du fait que l'usage de bonnes pratiques dans l'utilisation de la propriété @media permette de réaliser la même chose !

- Template Layout & Grid Alignment

  Il s'agit ici d'une référence au *CSS Grid Layout Module Level 1* dont la dernière évolution date de mars 2014 mais qui figure toujours au rang des brouillons (*Draft*) : *http://dev.w3.org/csswg/css-grid/.*

  L'idée est ici de travailler avec une grille découpée en lignes et colonnes définissant ainsi des zones dans lesquelles les informations vont être positionnées. En utilisant la propriété @media, il est possible de définir des caractéristiques différentes de la grille en fonction de l'orientation. Il est aussi introduit des notions d'unité proportionnelle comme le "fr" qui indique un rapport lié à l'espace libre. Par exemple, si l'ensemble des colonnes fait 900 px, mettre `grid-columns : 300 px 1fr` signifie que la première colonne est fixée à 300 px et le reste correspond à la fraction 1/1 du reste soit 600 px.

  Maintenant, si on met `grid-columns : 300 px 1fr 2fr`, alors la première colonne est toujours fixée à 300 px mais il y aura 2 autres colonnes dont la deuxième sera de 1/1+2 soit 1/3 de `600 px` et la troisième aura comme dimension 2/1+2 soit 2/3 de `600 px`...

  Au final nous aurons colonne n°1 : `300 px`, colonne n°2 : `200 px` et colonne n°3 : `400 px`.

  Bien sûr travailler avec des unités fractionnelles n'a de sens que si l'espace libre est inconnu. Dans le cas contraire il est plus facile de définir directement les tailles avec des unités de longueur non fractionnelles.

- Flexible Box Layout

  Le concept *Flexible Box Layout* fait référence à *CSS Flexible Box Layout Module Level 1* datant du 7 mars 2014. Celui-ci est d'ailleurs toujours à l'état de brouillon depuis 2009. La spécification *Flexible Box Layout* introduit en plus des 4 modes de

mise en page du CSS 2.1 (*inline, block, table* et *positioned*) un nouveau mode *flex* conçu pour des applications plus complexes.

Le *flex layout* ou *flexbox* vise à figer la mise en page, à distribuer les conteneurs dans un espace défini. Les contenus des conteneurs peuvent alors être affichés dans tous les sens (haut, bas, etc.), ils peuvent avoir un ordre d'affichage différent ou réarrangé, ils peuvent être linéaire collé à un seul axe d'affichage ou enveloppés autour d'un deuxième axe (en croix), leur taille peut s'adapter à l'espace disponible, ils peuvent être repliés ou non le long de l'axe principal sans intervenir dans la taille transversale du conteneur.

Ainsi la *flexbox* permet de mieux positionner des blocs enfants selon la surface d'affichage.

Voir l'article *http://www.alsacreations.com/tuto/lire/1493-css3-flexbox-layout-module.html* pour plus s'informations.

- Values and Units

La spécification en *release candidate CSS Values and Units Module Level 3* du 30 juillet 2013, définit les unités de valeur possibles et notamment :

- **les tailles relatives à la taille de la police** : *em* (par rapport à la taille héritée), *ex* (par rapport à la taille de la première police disponible), *ch* (taille par rapport à la hauteur du zero), *rem* (taille par rapport à la taille initiale de l'élément racine si celui-ci est définit avec font-size)
- **tailles en pourcentage de viewport** : *vw* (largeur de viewport), *vh* (hauteur de viewport), *vmin*, *vmax*
- **tailles fixes** : *pc* (12pt), *pt* (1/72 in), *px* (1/96 in), *in* (2,54 cm), *mm*, *cm*
- **tailles d'angles** : *deg* (1 cercle = 360 deg), *grad* (400 grad = 1 cercle), *rad* (1 cercle = 2∏ rad), *turn* (1 cercle = 1 turn)
- **tailles de résolutions** : *dpi*, *dpcm*, *dppx* (1 dppx = 96 dpi)
- **tailles calculées et autres expressions** : *calc()*, *toggle()*, *attr()*...

  Exemples :

  calc() est une fonction permettant de réaliser des calculs :

  ```
  width: calc(100%/3 - 2*1em - 2*1px);
  ```
  attention le résultat ne peut pas aller en dessous de zéro.

  toggle() pour la bascule des valeurs :

  ```
  em { font-style: toggle(italic, normal); }
  ```
  par exemple qui permet de faire passer l'italique, état normal de *em* en position normale si em se trouve à l'intérieur de texte déjà en italique ou

  ```
  ul { list-style-type: disc; }
  ul ul { list-style-type: toggle(disc, circle, square, box); }
  ```
  qui permet ici de définir 5 styles de puces en cascade lorsqu'elle sont imbriquées.

attr() est une fonction permettant d'attribuer les valeurs d'un attribut à un élément ou un pseudo élément :

```
width: attr(length em);
<stock>
<wood length="12"/>
<wood length="5"/>
<metal length="19"/>
<wood length="4"/>
</stock>
```

A priori, l'expression *calc()* serait similaire à *-epubx-exp*. Par contre, la règle *@-epubx-define* permet de réutiliser des valeurs de manière similaire à l'utilisation des variables personnalisées (*var-**) et leur réemploi avec la fonction *var()* tel que défini dans la spécification *CSS Custom Properties for Cascading Variables Module Level 1* du 7 mars 2014 mais toujours à l'état de brouillon.

Exemple :

```
:root {
  var-main-color: #06c;
  var-accent-color: #006;
}
/* The rest of the CSS file */
#foo h1 {
  color: var(main-color);
}
```

'*var-man-color*', introduit la variable nommée '*main-color*' et '*var-accent-color*' désigne la variable *'accent-color*'. La fonction *var('nom-de-ma-variable')* permet donc d'appeler le contenu d'une variable, ici le contenu de la variable '*main-color*', c'est à dire la valeur '*#06c*'.

- CSS Pagination Templates

Il est fait référence à *CSS Pagination Templates Module Level 3* qui reste à l'état de spécification non-officielle et brouillon, uniquement défendue par Adobe (*http://dev.w3.org/csswg/css-page-template/*). Il s'agit de travailler sur un concept de slots qui vont définir les caractéristiques des éléments... plus concrètement cela ressemble à l'identification de blocs dans l'espace d'affichage. D'ailleurs, cette notion "expérimentale" était présente dans la version Adobe Indesign CS6 lors du choix d'exportation EPUB. Sa disparition dans la version Adobe Indesign CC laisse entrevoir un abandon pur et simple de cette spécification.

- Paged Media

Il s'agit ici du brouillon du 7 mars 2013 *CSS Paged Media Module Level 3 (http://www.w3.org/TR/css3-page/)*, qui permet de définir la notion de *pagebox*, similaire au *viewport*, et ses caractéristiques (caractéristiques de la page de droite, de gauche etc.). Il s'agit aussi de définir comment présenter des contenus dans des séries de pages ou pour l'impression. Par exemple les contenus d'entête et pied de page vont être positionnés différemment ou contenir des choses différentes en fonction qu'ils apparaissent en page de gauche ou de droite (*CSS Generated*

Content for Paged Media Module de novembre 2011 à l'état de brouillon, *http://www.w3.org/TR/css3-gcpm/*) :

```
@page {
   @top-right { content: env(url) }
   @bottom-right { content: env(date-time) }
}
```

ou

```
@page { @top-center { content: string(header) }}
@page { @right-middle { content: string(index) }}
@page { @top-left { content: string(entry) }}
h1 { string-set: header "Chapter " counter(chapter) content()}
dt { string-set: index content(first-letter), entry content()}
```

2. Les interactions CSS - EPUB 3

Cette spécification *EPUB Adaptative Layout* ne rentre pas en conflit avec les appartenances CSS dans l'EPUB 3 et ne modifie pas les dispositions de mise en page et les caractéristiques de contenu déjà établies. En effet elle se contente de diffuser le contenu dans une séquence de conteneurs et de les placer sur la page. Cependant on note que des problèmes peuvent survenir dans le cas de blocs aux tailles non uniformes engendrés par le module *CSS Regions*.

Ceci jette à nouveau le doute sur l'emploi des *CSS Regions* comme il l'est mentionné plus haut avec le rejet de Google de supporter ce module dans son navigateur.

Concernant les valeurs *oeb-page-head* et *oeb-page-foot* de la propriété *display*, pour définir des contenus d'entête et pied de page, ceux-ci sont considérés comme figés et *-epubx-flow-into* prend les valeurs *oeb-page-head* et *oeb-page-foot*.

Toutes les propriétés ainsi que celles en @ doivent être affublées d'un préfixe tant que le W3C ne passe pas les spécifications à l'état de recommandation.

En règle générale, le préfixe *-epub-* est utilisé pour des éléments se basant sur des modules CSS stables alors que le préfixe *-epub**x**-* est employé lorsque les items dépendent de modules CSS instables ou directement spécifique à la spécification *EPUB Adaptative Layout* :

@-epubx-region
@-epubx-viewport
-epubx-flow-from
-epubx-flow-into
-epubx-shape-inside
-epubx-shape-outside
-epub-transform
-epub-transform-origin
-epubx-wrap-flow

3. Style adaptatif

Le style adaptatif permet de mettre en page les éléments en se servant d'une part de règles basées sur des conditions qui une fois réunies vont permettre au style de s'appliquer, et d'autre part sur le résultat de simples calculs.

Les valeurs calculées : -epubx-expr()

Pour mettre en place une mise en page complexe il faut s'appuyer sur les caractéristiques de hauteur et de largeur du média (*height, width*) et les préférences utilisateur comme la taille de police (*font size*).

En CSS 2.1, le seul moyen pour mettre en page dynamiquement les éléments consiste à se baser sur des unités en *pourcentage* et en '*em*'. Cependant ces moyens sont extrêmement limités puisqu'il est impossible par exemple de définir la taille de police de caractère par rapport à la hauteur d'un *viewport*. En outre il est impossible de définir des valeurs se basant sur des valeurs combinées comme par exemple définir une valeur par rapport à la fois, à la largeur d'un *viewpor*t et à la hauteur de celui-ci.

La fonction *-epubx-expr(<expression>)* applicable sur n'importe qu'elle propriété CSS, permet de réaliser des calculs et de renvoyer le résultat sous la forme d'une valeur.

Exemple :

```
.abspos {
      position:    absolute;
      width:       100px;
      height:      100px;
      left:        -epubx-expr(50% - 100px);
      top:         -epubx-expr(round((50% -
100px)/20px)*20px);
}
```

Les expressions acceptées sont :

- les expressions conditionnelles de type : arg ? arg2 : arg3 | arg4
- les fonctions de type : func(arg)
- les identifiant de classe : .class
- les préfixes : '-' et '!'
- les opérateurs logiques : '&&' et '||'
- les opérateurs de comparaison : '<','>', '=','==','<=','=>','!='
- les opérateurs mathématiques : '+', '-','*','/', '%'
- les unités: 'px', 'pc', 'pt', 'in','mm', 'cm','em','ex','%'

Il est recommandé de placer un espace après le signe '%' ainsi que pour le symbole '-' reconnu comme préfixe tout comme c'est le cas pour les expressions *XPath*.

Les unités valeurs sont toujours converties **en pixels** dans le contexte du document racine. Les pourcentages sont relatifs à la largeur du *viewport* exception faite des propriétés contenant les mots '*height*', '*top*', ou '*bottom*', qui font référence à la hauteur du *viewport*. Si la propriété recevant le résultat, est une mesure, alors le résultat de l'expression sera interprété en pixels. Dans le cas contraire, le résultat sera interprété comme une chaîne selon les règles de la propriété en question.

Le problème se pose lorsque des propriétés qui jusqu'alors n'acceptaient pas les nombres se mettent à les accepter subitement du fait des modifications d'une spécification, comme par exemple : *z-index, column-count*, et *opacity*.

Autre souci avec certaines propriétés comme :

- *page-width* et *page-height* : qui n'acceptent que des valeurs en pixels
- *true* et *false* : des valeurs booléennes

Les valeurs suivantes sont définies par défaut à false, 0, ou avec une chaîne vide, mais elles peuvent être remplacées par des valeurs utilisateur :

- *pref-font-family* : valeur de la famille de police pour <body>,
- *pref-night-mode* : [true] dans le cas où le mode nuit est choisi par l'utilisateur,
- *pref-night-contrast* : [true] pour le contraste fort,
- *pref-margin* : en pixels,
- *pref-line-height* : multiple de hauteur de caractère,
- *pref-column-width* : largeur de colonne en pixels,
- *pref-horizontal* : [true] dans le cas où l'horizontal serait choisi.

Liste des fonctions intrinsèques :

- floor(x) : le plus grand entier inférieur ou égal à x,
- ceil(x) : le plus petit entier supérieur ou égal à x,
- round(x) : arrondi à l'entier le plus proche,
- sqrt(x) : racine carrée,
- min(x,y) : donne la plus petite valeur entre x et y,
- max(x,y) : donne la plus grande valeur entre x et y,
- letterbox(viewW, viewH, objW, objW) : permet la mise à l'échelle d'un objet,
- typeof(x) : donne le typage.

■ Les variables personnalisées : @-epubx-define

La règle *@-epubx-define* permet de définir un certain nombre de variables personnalisées et d'y affecter des valeurs pour les réutiliser selon la syntaxe suivante :

```
@-epubx-define { [<name> : <value>;]*}
```

'*name*' doit être une expression CSS valide tout comme '*value*' qui peut accepter le résultat de la propriété *-epubx-expr()*.

La variable ne peut être utilisée qu'une seule fois à l'intérieur des feuilles de styles référencées dans un ouvrage. Elle ne peut pas être redéfinie mais peut changer pour souligner d'autres caractéristiques. Elle peut aussi être utilisée pour définir d'autres valeurs personnalisées. L'ordre n'a aucune importance et une valeur peut être appelée avant qu'elle ne soit nommée du moment qu'il n'y a pas de références circulaires. Si une variable contient une référence circulaire, alors elle sera évaluée comme si elle contenait une chaîne vide.

Il est fortement découragé de redéfinir une variable sauf dans le cas où il convient d'adopter les préférences de l'utilisateur.

Exemple :

```
@-epubx-define {
full-page-width: -epubx-expr(max(page-height*4/3,page-width));
}
```

Le style conditionnel : @-epubx-when

L'expression *@-epubx-when* permet de définir des styles si certaines conditions sont réunies via la syntaxe suivante :

```
@-epubx-when <expression> { [<css-rule>]*}
```

L'expression doit être valide sous les mêmes conditions que pour *-epubx-expr()*. Le style conditionnel doit être utilisé dans les cas où il n'est pas possible d'utiliser le *Media Queries* pour réaliser l'opération ou bien lorsqu'il doit être mentionné des variables personnalisées (*-epubx-define*).

```
@-epubx-when page-width*page-height > 800000 {… }
```

4. Les pages

Flux, partition et pages maîtres

Le **flux** est une séquence d'éléments affichés dans un ordre donné tout comme un article dans un journal. A l'origine un document est considéré comme n'ayant qu'un seul flux appelé « *body* » qui contient l'élément racine du document. Les éléments peuvent être assignés à d'autres flux en spécifiant les propriétés de -

epubx-flow-into du *CSS Regions*. Les flux sont alors créés selon le besoin, c'est à dire dès qu'un élément lui est assigné.

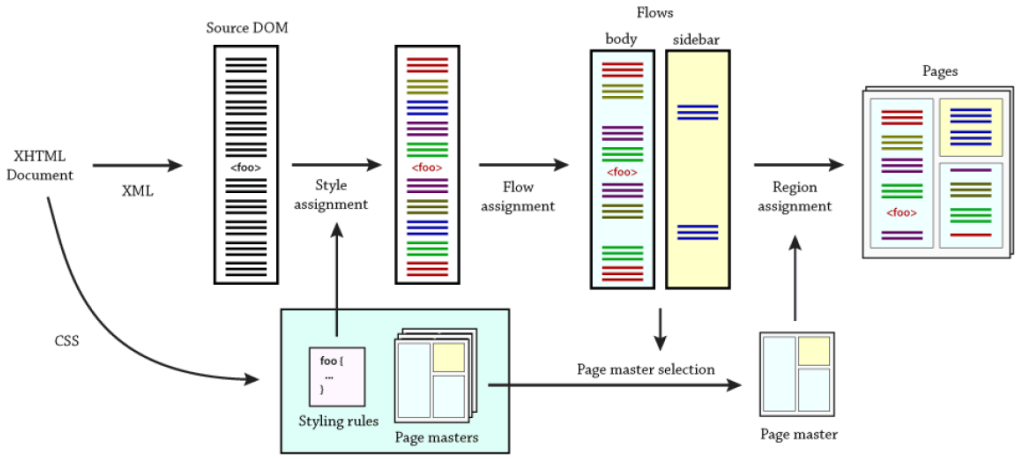
Une **partition** est une zone de la page qui permet d'afficher le contenu d'un flux particulier. Dans la plupart des cas, il y a plusieurs partitions dans une séquence de pages ou sur une seule page permettant d'afficher la totalité du contenu d'un flux particulier.

Le contenu est dit « **coulé** » (*to flow*) d'une partition à une autre partition, sur le même principe des vases communicants.

Les partitions peuvent contenir plusieurs colonnes. Elles peuvent s'adapter au contenu en s'agrandissant dans certaine mesure ou être de taille fixe, et elles peuvent aussi adopter des dimensions différentes des unes des autres. D'ailleurs, les partitions peuvent très bien avoir des colonnes de tailles différentes pour s'adapter à un même contenu. Pour définir une partition il faut utiliser la propriété *@-epubx-partition* ; la propriété *-epubx-flow-from* permettra, quant à elle, d'affecter un flux à une partition.

Attention, dans les spécifications W3C des *CSS Regions*, la notion est plus générale, une région peut être créée à l'intérieur de n'importe quel élément ou pseudo élément. Dans ce contexte les régions ne peuvent être créées qu'à l'extérieur des partitions.

Une **page maître** est une allocation de page à l'intérieur d'un jeu de partitions. Les pages maîtres permettent aux auteurs de présenter de manière sophistiquée les contenus en contrôlant la géométrie des multiples partitions d'une page. C'est une sorte de gabarit ou modèle de page. Une feuille de style peut spécifier n'importe quel nombre de pages maîtres, et une page maître particulière est choisie pour chaque page algorithmique. Les pages maîtres sont définies en utilisant la règle *@-epubx-page-master*.



Syntaxe et sémantique

La spécification CSS Adaptative Layout utilise la même syntaxe et sémantique que la spécification CSS Regions (*http://www.w3.org/TR/2012/WD-css3-regions-*

20120503/) et CSS Exclusions (*http://www.w3.org/TR/2012/WD-css3-exclusions-20120503/*) mais y affecte un préfixe *-epubx-* :

@-epubx-**region**
-epubx-**flow-from**
-epubx-**flow-into**
-epubx-**shape-inside**
-epubx-**shape-outside**
-epubx-**wrap-flow**

Pour le multicolonnage il n'y a pas besoin de préfixe et donc l'intégralité du *CSS Multi-column Layout Module* peut s'y appliquer (*http://www.w3.org/TR/css3-multicol/*).

Pour les transformations, il est nécessaire d'utiliser le préfixe *-epub-* et la syntaxe retenue est celle du CSS 2D Transforms (*http://www.w3.org/TR/2011/WD-css3-2d-transforms-20111215/*) tandis que la sémantique est celle du CSS Transforms Module Level 1 (*http://www.w3.org/TR/css3-transforms/*) :

-epub-**transform**
-epub-**tranform-origin**

Pour les fonds et les bordures :

border-color

border-style

border-width

border-radius

background-color

background-image

background-position

background-size

background-repeat

background-origin

background-clip

background-attachment

box-shadow

box-decoratiion-break

Style du contenu

1. *Le flux*

Le Flux représente un regroupement logique d'un ou plusieurs éléments d'un document source dans un débit unique de contenu. Un flux est créé dès qu'un élément est associé à un flux nommé à l'aide de la propriété *-epub-flow-into*.

Les éléments ajoutés au flux n'ont pas besoin d'être contigus ni avoir leur localisation dans le DOM HTML. C'est pour cette raison que les éléments ajoutés au flux ne modifient pas la structure du document et que tous les styles appliqués aux éléments demeurent en place.

Si des éléments sont affectés à plusieurs flux, alors il y aura autant de séquences que de flux. Pour chaque flux, le contenu est ensuite placé au fil de l'eau tant qu'il existe des partitions disponibles. Les partitions sont identifiées grâce à la propriété *flow-from*. Par défaut les éléments sont disposés dans l'ordre où ils apparaissent dans le flux.

Lorsque l'espace d'une partition est rempli, le contenu restant est déversé dans la partition suivante, à l'instar des vases communicants. Cependant lorsqu'un élément n'est pas complètement déversé dans la partition, alors celui-ci est décomposé de la même manière qu'ils le sont habituellement dans les colonnes et les pages : la partie restante est reportée dans la partition suivante. Ce fonctionnement par défaut peut être altéré par des propriétés assignées au contenu à l'intérieur du flux.

Un contenu qui ne serait pas assigné à un flux particulier, appartient au flux à *body*.

Exemple de processus

```
.sidebar { -epubx-flow-into: leftbox;}

@-epubx-page-master {
          @-epubx-partition {
                -epubx-flow-from: body;
                column-width: 25em;
          }

          @-epubx-partition {
                -epubx-flow-from: leftbox;
                height: 6em; width: 6em;
          }
}
```

Propriétés	Description
-epubx-flow-options	La propriété s'applique aux éléments pour lesquels une propriété *-epubx-flow-into* a été spécifiée. **Valeurs :** **none** (par défaut) | exclusive || last || static Les valeurs peuvent être combinées (*exclusive last*, *exclusive static*). **exclusive** : marque l'élément qui est alors considéré en premier. Il ne peut y avoir qu'un seul élément 'exclusive' dans une unique partition, et un élément est toujours absorbé et retiré du flux une fois l'affichage généré à moins qu'il ne soit aussi marqué comme 'static'. **static** : en marquant un élément comme 'static', cet élément au lieu d'être consommé et retiré du flux, celui-ci est replacé dans le flux et donc va être déversé dans la partition suivante. **last** : combiné à '*exclusive*', l'élément ainsi marqué va être déversé dans la partition suivante. En fait dans une série d'éléments marqués 'exclusive', celui qui possèdera en plus l'attribut 'last' ira se placer dans la partition suivante disponible.
-epubx-flow-linger	La propriété s'applique aux éléments pour lesquels une propriété *-epubx-flow-into* a été spécifiée. **Valeurs :** **none** |nombre entier **(nombre de pages)** Permet de définir le nombre de pages éligibles pour un document. En définissant la valeur à '*none*', l'élément reste disponible tant qu'il n'est pas consommé. En définissant la valeur à 1, l'élément peut être utilisé sur la même page sur laquelle il est apparu.
-epubx-flow-priority	La propriété s'applique aux éléments pour lesquels la propriété *-epubx-flow-into* a été spécifiée et qui possèdent en plus la valeur '**exclusive**' pour la propriété *-epubx-flow-options*. **Valeurs :** **0** (par défaut) | nombre entier Cette propriété permet de définir l'ordre dans une séquence d'éléments 'exclusive'. L'élément possédant la priorité la plus forte est sélectionné.

Les propriétés *break-inside*, *break-before*, *break-after* peuvent s'appliquer pour une partition possédant des colonnes multiples. Néanmoins les valeurs *inside*, *before* et *after* pourront s'appliquer à des partitions dénuées de colonnes.

2. Pages maîtres et partitions

Les pages maîtres sont déclarées avec la règle *@-epubx-page-master* et elles peuvent être affublées d'un nom ou d'un nom de classe (si déclaré en entête) pour être plus facilement réutilisées. A l'intérieur de *@-epubx-page-master*, il est possible de déclarer une partition ou un groupe de partitions par le biais de *@-epubx-partition* et *@-epubx-partition-group*. Le minimum étant d'en déclarer au moins une pour le group de partition et pour la partition.

```
@-epubx-page-master [ <name>] ? [ class(<name>) ]* {
[ <propname>: <propvalue>; | <epubx-partition> | <epubx-
partitiongroup>]*
}
```

Les propriétés suivantes peuvent s'appliquer :

-epubx-page (pour la position de la page)

-epubx-enabled (disponibilité de la page)

-epubx-min-page-height (dimension minimale de la hauteur de page)

-epubx-min-page-width (dimension minimale de la largeur de page)

-epubx-snap-height

-epubx-snap-width

-epubx-required

Les pages maîtres peuvent aussi inclure les propriétés '*opacity*' et '*background*' seulement les propriétés '*background-attachement*', '*background-clip*', et '*background-origin*' seront ignorées.

Dans le processus il est considéré que si une page maître est disponible alors elle peut être utilisée. Toute la complexité du problème consiste donc à déterminer si une page maître est disponible ou non.

Une fois la page maître définie, il convient de définir la ou les zones (partitions) à l'intérieur de la page en utilisant les propriétés *@-epubx-partition* qui peuvent comme la page maître, être affublées d'un nom ou d'un nom de classe en utilisant la même syntaxe.

```
@-epubx-partition [ <name> ]? [ class(<name>) ]* {
[ <propname>: <propvalue>;]*
}
```

Les propriétés utilisables dans les partitions sont :

-epubx-flow-into, top, left, bottom, right, width, height, margin, border, padding, background, z-index, overflow, opacity, cursor, columns, column-gap, column-rule, border-radius, box-shadow, -epub-transform, -epub-transform-origin, -epub-writing-mode, -epubx-shape-inside, -epubx-shape-outside et -epubx-wrap-flow.

Dans le cas des partitions devant afficher un contenu à l'horizontal, la hauteur de celle-ci est définie par son contenu à moins que la propriété *height* ne soit déjà définie ou spécifié à '*auto*'. De la même manière les partitions devant afficher du contenu à la verticale (propriété *-epub-writing-mode*) vont voir leur propriété affectée.

Remarque :

- La propriété *-epubx-writing-mode* ne peut pas recevoir une valeur en provenance de *-epubx-expr()*.
- La propriété '*overflow*' pour une partition est définie par défaut à '*hidden*' cependant les utilisateurs peuvent très bien l'écraser en la définissant à '*visible*', mais aucune autre valeur ne sera admise.
- La propriété *-epubx-wrap-flow* d'une partition est définie par défaut à '*both*' ceci entraîne des exclusions de contenu pour toutes les autres partitions adjacentes. Cependant il est possible de contrer cette propriété en lui donnant la valeur '*auto*' ce faisant le contenu débordera sur les autres partitions et en annulant les autres paramètres.
- Les partitions ne comportent pas obligatoirement les contenus des partitions antérieures.
- Le contenu est automatiquement exclu pour les partitions qui dépendent de leur progression : si une partition voit sa taille grandir automatiquement par calculs et que sa position dépend de ce calcul alors les exclusions ne seront pas appliquées.

Les propriétés suivantes du '***CSS Adaptative Layout***' sont applicables :

-epubx-enabled, -epubx-required, -epubx-required-partitions, -epubx-conflicting-partitions, -epubx-min-page-height, -epubx-min-page-width, -epubx-page, -epubx-snap-with, -epubx-snap-height.

Le simple fait d'associer la propriété *-epubx-flow-from* à une partition, lie la partition au contenu et le contenu du flux est déversé dans la partition. Par défaut *-epubx-flow-from* est défini pour '*body*'.

Exemple :

```
@-epubx-page-master {
    @-epubx-partition body {
            -epubx-flow-from:body;
            column-width: 20em;
            padding: 0.5em;
            margin: 1em;
            border: 2px solid black;
    }
}
```

Le contenu va être déversé dans une page comportant padding, marge, bords et multicolonnage.

```
@-epubx-page-master {

    @-epubx-partition body {
           -epubx-flow-from:body;
           column-width: 20em;
           margin: 1em;
    }

    @-epubx-partition head {
           -epubx-flow-from:head;
           margin: 1em;
           border-bottom: 1px solid black;
    }
}
```

Dans le cas présent, si la partition '***head***' recoit du contenu d'un flux '***head***' alors le contenu dans la partition '***body***' ne sera pas déversé.

- **Utilisation du CSS Regions pour l'application de style à l'intérieur de partitions : @-epubx-region**

Voici un exemple où le contenu d'une région de haut de page '*top*' situé dans une colonne unique est déversé dans une région de bas de page '*bottom*' possédant 2 colonnes.

```
@-epubx-page-master {
       @-epubx-partition class(top) {
              -epubx-flow-from:body;
              height: 6em;
       }

       @-epubx-partition class(bottom) {
              top: 7em;
              -epubx-flow-from:body;
              column-count: 2;
       }
}

@-epubx-region .top {
       p { font-size: 1.1em ; }
}
```

A noter que les partitions n'ont pas de relation ascendant-descendant et ainsi le style suivant n'aurait aucun effet :

```
.top p {font-size: 1.1em ;}
```

Les groupes de partitions @-epubx-partition-group

En général, les règles de groupe *@-epubx-partition-group* sont emboîtées à l'intérieur de la déclaration *@-epubx-page-master*. Néanmoins il peut être nécessaire de posséder un conteneur commun. En effet, appliquer des propriétés sur des sous-conteneurs individuels pourraient avoir des résultats surprenants comme cepeut être le cas pour les propriétés '*z-inde*x" et '*opacity*'.

Dans d'autres cas il peut simplement être plus confortable de posséder une *CSS Box* contenant plutieurs partitions.

La déclaration *@-epubx-partition-group* peut être utilisée pour grouper les partitions. Attention, le fait de grouper des partitions entre-elles, ne modifie pas leur portée, leur position ou leur comportement particulier.

@-epubx-partition-group peut contenir elle-même des déclarations *@-epubx-partition* et *@-epubx-partition-group* et définir certaines propriétés :

- opacity
- -epub-transform
- -epub-transform-origin
- visibility
- overflow
- z-index
- -epubx-snap-width
- -epubx-snap-height

Syntaxe :

```
@-epubx-partition-group [<name>]? [class(<name>)]* {
        [<propname> : <propvalue>;
        | <epubx-partition>
        | <epubx-partition-group>]*
}
```

Les propriétés sont optionnelles mais toute groupe de partition *@-epubx-partition-group* doit contenir au minimum soit une déclaration de partition *@-epubx-partition* ou soit une déclaration de groupe de partition *@-epubx-partition-group*.

Utilisation des pages maîtres conditionnelles avec la propriété : -epubx-enabled

Dans la plupart des cas, une page maître ou une partition peut être utilisée si certaines conditions sont réunies. Avant qu'une page maître ne puisse être utilisée (ou une partition à l'intérieur d'une page maître sélectionnée), il convient de déterminer si elle est activées '*enabled*' ou non en utilisant la propriété *-epubx-enabled*. Le mode de détermination du paramètre est différent qu'il s'agisse d'une page maître ou d'une partition.

Cependant dans tous les cas, la partition ou la page maître sera déclarée comme activée (paramètre '*true*') lorsque les conditions suivantes sont remplies :

soit la propriété *-epubx-enabled* n'est pas spécifiée soit si elle existe sa valeur est *'true'*.

- soit la propriété *-epubx-min-page-width* n'est pas définie soit la largeur de la page est plus grande ou égale à *-epubx-min-page-width*.
- soit la propriété *-epubx-min-page-height* n'est pas définie soit la hauteur de la page est plus grande ou égale à *-epubx-min-page-height*.
- soit la propriété *-epubx-page* n'est pas définie ou soit l'un des numéros de page d'une des pages produite est égale à *-epubx-page*.

Pour les partitions, des propriétés supplémentaires doivent être remplies. Pour information, la partition est considérée avoir une hauteur explicite lorsque sa hauteur '*height*' n'est pas définie à '*auto*' ou bien lorsque la valeur des deux propriétés '*top*' et '*bottom*' n'est pas définie à '*auto*'.

- '*required*' n'est pas définie ou bien il existe du contenu disponible dans le flux correspondant.
- toutes les partitions référencées dans -epubx-required-partitions sont activées.
- aucune des partitions référencées par *-epubx-conflicting-partitions* n'est activée.

Pour les pages maîtres :

- toutes les partitions pour lesquelles la propriété *-epubx-required* est spécifiée à '*true*' doivent être activées.

Utilisation de mise en page spécifique avec la propriété: -epubx-page

La mise en page d'une page maître ou une partition peut s'appliquer à une page particulière lorsque la propriété -epubx-page (de la page maître ou la partition) indique le numéro de la page concernée.

Sélectionner une page maître en se basant sur les dimensions du support, propriété: -epubx-min-page-width et -epubx-min-page-height.

Les propriétés *-epubx-min-page-height* et *-epubx-min-page-width* définissent les dimensions minimales pour chaque page maître ou partition encore disponibles. Elles partiticipent à la détermination de la propriété *-epubx-enabled*.

Sélectionner une page maître en se basant sur un contenu : la propriété -epubx-required

La propriété *-epubx-required* détermine si une partition doit être présente sur une page maître pour être utilisable. Cette propriété convient particulièrement pour sélectionner une page maître en se basant sur la disponibilité du contenu dans un flux particulier.

Cette propriété participe à la détermination de la propriété *-epubx-enabled* interne à la page maître.

- **Utilisation ou non de partitions associées : propriété -epubx-conflicting-partitions**

Cette propriété détermine si une partition ne doit pas être utilisée conjointement à une ou plusieurs partitions sur la page maître. Par exemple c'est le cas, si l'utilisation d'une partition, conjointement à l'utilisation d'autres partitions de la page maître, cause des chevauchements ou des conflits d'affichage.

Elle participe à la détermination de la propriété *-epubx-enabled* pour la page maître dans laquelle elle est intégrée.

Attention, elle ne peut pas prendre une valeur issue d'une expression calculée par *-epubx-expr()*.

- **Utilisation de plusieurs partitions conjointes : propriété -epubx-required-partitions**

A l'inverse de la propriété précédente, il s'agit ici de déterminer si la partition doit être utilisée uniquement conjointement à l'utilisation d'une ou plusieurs partitions spécifiques sur la page maître. Elle participe à la détermination de la propriété *-epubx-enabled* pour la page maître dans laquelle elle est intégrée.

Attention, elle ne peut pas prendre une valeur issue d'une expression calculée par *-epubx-expr()*.

- **Accrochage à une ligne de la grille : les propriétés -epubx-snap-height et -epubx-snap-width**

Ces propriétés n'ont d'effet que sur les déclarations @-epubx-partition cependant si elles sont spécifiées à l'intérieur d'une déclaration @-epubx-partition-group ou @-epubx-page-master, la valeur est héritée dans la déclaration @-epubx-partition.

Dans le cas de mise en page pour le print, les lignes de texte à l'intérieur de colonnes adjacentes sont alignées. Ce type d'alignement peut être réalisé dans une mise en page de type CSS multicolumn s'il est accordé une attention particulière à l'espacement de paragraphe et la hauteur de ligne. En effet chaque chaque colonne débute à la même position dans le bloc de progression.

Quoiqu'il en soit si le texte est flottant à l'intérieur de partitions positionnées arbitrairement, les lignes vont être alignées seulement si les partitions sont positionnées sur une nombre de lignes en dehors du bloc de progression. Les propriétés *-epubx-snap-height* et *-epubx-snap-width* vont aider à contrôler le positionnement de manière automatique.

La propriété *-epubx-snap-height* définit la direction verticale et donc est très utile pour l'écriture horizontale tandis que *-epubx-snap-width* définit la direction horizontale et donc sera utile pour l'écriture verticale. Les deux propriétés peuvent être utilisées conjointement sur la même partition ou la même page maître.

Ces propriétés enclenchent la position d'origine du contenu et les bords des partitions flottant qui chevauchent la grille, en adoptant la même intervalle que pour la propriété '*line-height*' définie pour le contenu de la partition. La position de la partition est poussé dans le bloc ou progresse dans la même direction 'inline' de la grille suivante.

Les bords de l'exclusion sont déplacés pour agrandir l'exclusion, comme pour le bord supérieur et vers le bas pour le bord inférieur. Les partitions qui ont causé elles-mêmes les exclusions ne sont pas déplacés, l'ajustement ne se produit que pour les zones d'exclusion qu'elles génèrent.

■ Les flux primaires : @-epubx-flow et -epubx-flow-consume

La déclaration *@-epubx-flow* permet d'établir les propriétés globales du flux. Cependant elle ne le crée pas, il suffit d'utiliser la propriété *-epubx-flow-into* pour créer le flux.

Syntaxe :

```
@-epubx-flow <name> {
        [<propname>: <propvalue>;]*
}
```

Au plus il ne peut y avoir qu'une seule déclaration pour un type de flux nommé. La seule propriété possible à l'intérieur d'une déclaration est *-epubx-flow-consume*.

■ Définir un champ de recherche : -epubx-utilization

Tout le principe de sélection de page maître repose sur le fait que le contenu admissible à la page suivante est connu à l'avance.

Le contenu "admissible" (décrit en détail dans la section suivante) se base sur la capacité du document à anticiper la façon avec laquelle le contenu va tenir sur la page suivante.

Cette anticipation est une simple heuristique et dans certains cas, il est souhaitable de l'ajuster. Par exemple, si la page maître contient la partition du

flux primaire qui couvre seulement une petite fraction de la zone de page, il peut être souhaitable de diminuer la taille d'anticipation.

La propriété *-epubx-utilization* définit une gamme de contenu qui est recherché avant de trouver le contenu disponible pour différentes partitions.

La zone de recherche est déterminée de la manière suivante : la zone de la page est mesurée '*em*' carrés (l'aire d'un carré de longueur de côté égale à 1em), multiplié par la valeur *-epubx-utilization* puis arrondi.

Le résultat est la quantité de contenu à attendre pour la page maître. La position de recherche dans le contenu est déterminé par :

Le départ de la position courante dans le document.

L'initialisation de la variable du contenu restant en amont du contenu à déterminer en aval.

L'itération se poursuit élément par élément et nœud par nœud UTF-16, les éléments sont considérés avant leurs enfants

Passer éléments pour lesquels la propriété 'display' est réglée sur 'none' ainsi que tous les éléments qui n'appartiennent pas au flux primaire. ainsi que leurs descendants, sinon,

Pour chaque point codé UTF-16 dans un nœud de texte en soustraire 1 du contenu restant.

Pour chaque élément soustraire 1 élément du contenu restant.

Position est atteinte lorsque le contenu restant devient nul ou que la fin du document est atteinte.

A noter que la position de recherche peut apparaître entre deux points. Ce n'est pas un problème, car il n'est jamais utilisé pour diviser le contenu textuel.

Les expressions à l'intérieur de @-epubx-page-master

Chaque déclaration *@-epubx-page-master* crée une étendue pour les valeurs nommées. En plus des noms définis par la déclaration *@-epubx-define*, il est possible d'accéder à certaines valeurs des partitions de la page maître en utilisant le point.

Syntaxe :

```
<partition name>.<value-name>
```

ou

```
<partition name>.<function-name>(arg...)
```

En accédant à la valeur d'une propriété par son nom, la valeur renvoyée est celle du moment. Par exemple pour la propriété de type '*width*', la valeur de *'width*'

du partition sera celle qui sera établie après l'application de toutes les règles CSS.

Concernant les noms dans les formulaires avec la syntaxe de type ***<box-name><side>-bord*** où ***<box-name>*** peut prendre les valeurs : *<empty>*, *padding-*, *border-*, *margin-*. Et *<side>* peut prendre les valeurs *top-*, *bottom-*, *left-*, *right-*.

Ces noms correspondent aux positions des bords des boxes relatifs au coin haut-gauche de la page. *<empty>* correspond à la surface du contenu. La seule fonction utilisable est *columns(count)* :

```
min(count, column-count)* (column-width + column-gap) -
column-gap
```

Habituellement, les valeurs peuvent être utilisées avant d'être définies aussi longtemps qu'il n'y a pas de référence circulaire. Les règles permettant de calculer les valeurs actuelles des propriétés d'une partition sont incluses implicitement dans le mode global de calcul des valeurs et de la détermination de la résolution. La seule contrainte est que lorsque la hauteur d'une partition n'est pas spécifiée ou bien définit à 'auto' (ou la largeur dans le cas de l'écriture verticale), les règles CSS prescrivent de disposer le contenu afin de déterminer la hauteur de bloc.

Les partitions sont toujours disposées dans l'ordre inverse de leur parution dans la page maître. Ainsi les paramètres de la partition peuvent simplement dépendre de la hauteur des partitions qui suivent ou de partition dont la hauteur est explicitement définie.

Exemple :

```
@-epubx-page-master {
       @-epubx-partition body {
          -epubx-flow-from: body;
          column-width: 20em;
       }

       @-epubx-partition pict {
          -epubx-flow-from: pict;
          left: -epubxexpr(body.columns(body.columncount - 1) -
          pict.width/2);
          top: -epubxexpr((page-height - pict.height)/2);
          width: -epubxexpr(min(300px,20em));
       }
}
```

Dans cet exemple la partition '*pict*' est positionnée horizontalement entre les deux colonnes de droites de la partition '*body*' et centrée dans la page verticalement.

3. Tableau récapitulatif

Propriétés	Description
-epubx-enabled	La propriété s'applique à *@-epubx-page-master* et *@-epubx-partition.* Valeurs : **true** (par défaut) | false
-epubx-page	La propriété s'applique à *@-epubx-page-master* et *@-epubx-partition.* Valeurs **:** entier
-epubx-min-page-height	Donne la hauteur minimale d'une partition ou page maître. La propriété s'applique à *@-epubx-page-master* et *@-epubx-partition*. Valeurs : entier, valeur initiale : 0
-epubx-min-page-width	Donne la largeur minimal d'une partition ou page maître. La propriété s'applique à *@-epubx-page-master* et *@-epubx-partition.* Valeurs : entier, valeur initiale : 0
-epubx-required	Définit si une partition est requise sur une page maître pour être utilisable. Elle s'applique à *@-epubx-partition.* Valeurs : **true** (par défaut) | false
-epubx-conflicting-partitions	Elle s'applique à *@-epubx-partition* et détermine les partitions pour la/lesquelles elle rentre en conflit et pour la/lesquelles il ne faut pas l'utiliser. Valeurs : chaîne de nom de partition(s) séparée(s) par des virgules, vide par défaut.
-epubx-required-partitions	Elle s'applique à *@-epubx-partition* et détermine si la partition en question doit être utilisée conjointement à l'utilisation de certaines partitions sur la page maître. Valeurs : chaîne de nom de partition(s) séparée(s) par des virgules, vide par défaut.
-epubx-snap-height	Elle s'applique à *@-epubx-partition* et détermine la façon d'accrocher les lignes de textes sur la verticale. La propriété est héritée. Valeurs : **none** (par défaut) | hauteur
-epubx-snap-width	Elle s'applique à *@-epubx-partition* et détermine la façon d'accrocher les lignes de textes sur l'horizontale. La propriété est héritée.

	Valeurs : **none** (par défaut) | largeur
-epubx-flow-consume	S'applique à *@-epubx-flow*. Valeurs : **all** (pour le flux body) | **some** (pour les autres flux) Si la propriété est définie à '*all*' pour un flux particulier, alors celui-ci est considéré comme primaire. Les pages sont générées à partir de la page maître tant qu'un flux primaire a du contenu à déverser. Une fois que le contenu du flux primaire est déversé, le document est considéré comme complet.
-epubx-utilization	Elle s'applique à *@-epubx-page-master* et permet d'établir le nombre de d'éléments de contenu qui va être recherché pour déterminer le contenu disponible pour les partitions. Valeurs : 1 (par défaut) | nombre entier

4. Modèle de traitement de la mise en page

1. La propriété *-epubx-flow-into* de chaque élément du document est déterminée
2. Les éléments qui appartiennent au même flux sont rassemblés en séquences.
3. Les déclarations *@-epubx-flow* sont examinées et les flux notés *-epubx-flow-consume:all* sont marqués en tant que flux primaire. Le flux '*body'* est primaire par défaut.

A noter que chaque élément est toujours au courant de sa position dans le document. En d'autres termes, les séquences de flux contiennent des références à des éléments dans le document, et non pas les éléments eux-mêmes.

Chaque élément utilisé dans l'algorithme de pagination possède les valeurs suivantes qui leur sont associés :

- **statique (booléen) - true** si l'élément est marqué par *-epubx-flow-options : static*
- **exclusif (booléen)** - **true** si l'élément est marqué par *-epubx-flow-options : exclusive*
- **last (booléen) - true** si l'élément est marqué par *-epubx-flow-options : last*
- **priority (entier)** - Valeur de la propriété *-epubx-flow-priority*
- **linger-count (integer | infinity)**, valeur initiale : **infinity**
- **consumed (fully|partially|not)**, valeur initiale : **not**

- **consumed-offset** (position du document), initialement au début de l'élément.

Pour produire la page suivante, procéder comme suit :

1. Déterminer la position actuelle dans le document en trouvant la valeur minimal de la propriété *consumed-offset* pour tous les éléments non entièrement consommés dans chaque flux primaire. La position courante est le montant maximum de résultats de flux primaires.
2. Pour la sélection de page maître, pour chaque page maître :
   - Calculer la position de recherche en utilisant la position courante et son utilisation (voir *-epubx-utilization*)
   - Déterminer l'élément éligible. Chaque élément du flux est considéré comme éligible s'il n'est pas marqué comme pleinement consommé et il arrive dans le document avant la position de recherche.
   - Déterminer la disponibilité du contenu. Un flux possède du contenu disponible s'il contient du contenu éligible.
   - Déterminer si la page maître est active en utilisant la propriété *-epubx-enabled*.
   - La première page maître active est utilisée pour la page suivante.
3. Pour chaque élément qui devient éligible sur la page, affectez sa propriété *linger-count* de la valeur de la propriété *-epubx-flow-linger*.
4. Passer en revue les partitions dans la page maître dans l'ordre inverse, et pour chaque partition trouver son flux correspondant et positionner le contenu en utilisant les principes suivants (seuls les éléments éligibles sont concernés) :
   - **A**. S'il n'y a des éléments exclusifs dans le flux, considérer que les éléments exclusifs. Choisir les premiers avec une priorité la plus forte. Sinon ne considérer que ceux qui sont notés '*last*'. Si plusieurs sont notés '*last*', choisir le dernier. Si aucun élément n'est noté '*last*', prendre le premier élément, puis placez le dans la partition. S'il déborde les débordements seront gérés par les règles CSS normales. Marquer cet élément comme entièrement consommé.
   - **B**. S'il n'y a pas d'éléments exclusifs, placer le premier élément non entièrement consommé débutant à partir de son décalage (*consumed-offset*). Lorsque la teneur de cet élément est épuisée, il est marqué comme pleinement consommée puis l'élément non entièrement consommé suivant dans le flux est ensuite placé. Lorsque la partition est entièrement remplie, le dernier élément incomplet placé est réparti de la même manière que les éléments sont répartis dans les colonnes et les pages ; l'élément est alors marqué comme partiellement consommé et son décalage (*consumed-offset*) est réglé au point de rupture.

5. Réinitialiser l'état de consommation de tous les éléments statiques en les redéfinissant à 'non consommé' et repositionnez leur décalage (*consumed-offset*) au début de l'élément.
6. Soustraire 1 de la valeur de la propriété *linger-count* de tous les éléments admissibles (à moins que celle-ci ne possède la valeur '*infinity*'). Marquez les éléments possédant une valeur *linger-count* négative comme étant entièrement consommés.

5. Le contrôle du viewport

La définition du viewport avec la déclaration @-epubx-viewport

Les propriétés déclarées dans @-epubx-viewport s'appliquent à l'ensemble de la mise en page de la page.

Syntaxe :

```
@-epubx-viewport {
        [<propname>:<propvalue>;]*
}
```

Les propriétés suivantes sont autorisées :

- *width*
- *height*
- *-epubx-text-zoom*

Attention néanmoins, les propriétés 'height' et 'width' doivent absolumment être utilisées conjointement. Une fois renseignées les deux valeurs définissent alors la taille logique du viewport. Toutes les pages auront alors la même taille telle que définie en viewport et seront zoomées ou dézoomées pour tenir dans la taille du viewport.

Le contrôle du zoom dans le viewport avec la propriété -epubx-text-zoom

La propriété *-epubx-text-zoom* permet de contrôler comment le contenu va ête zoomé ou plus approximativement l'augmentation de la taille de caractère. Elle n'est utilisable qu'à l'intérieur de la déclaration *@-epubx-viewport* et ne peut prendre que 2 valeurs : '*font-size*' et *'scale*'.

Lorsque la propriété prend la valeur par défaut '*font-size*', alors le zoom va agrandir la taille initiale des caractères si tant est que celle-ci soit définie en '*em*'. Dans tous les cas, le zoom augmente la taille de tous les éléments dont l'unité est exprimée en '*em*' ou '*ex*'. Si la valeur de *-epubx-text-zoom* est définie à '*scale*' alors la taille logique du viewport est dézoomée

proportionnellement ce qui a pour conséquence de dézoomer l'ensemble des pages dans les mêmes proportions.

On peut se demander si les informations définies dans les déclarations *@-epubx-viewport* et *-epubx-text-zoom* ne vont pas rentrer en contracticton ou en conflit avec les caractéristiques du viewport définies pour chaque page dans la balise meta (viewport et initial-scale) par exemple :

```
<meta
       name=          "viewport"
       content=       "width=device-width,initial-scale=1"
/>

// ou sont pendant CSS :

@viewport {
    width:    device-width;
    zoom:     1;
}
```

6. Compatibilité avec la règle @-epubx-page-template

Toutes les constructions définies dans cette spécification doivent être contenues à l'intérieur d'une déclaration *@-epubx-page-template* afin de garantir un compatibilité avec les applications de lecture qui ne les supporteraient pas. La détermination de l'activation ou non des propriétés liées à cette spécification n'est pas encore établie mais il est probable que celle-ci ne soit pas activée pour les écrans de très petite taille (sauf pour l'impression) ou pour les supports oraux.

Bien sûr si les spécifications ne sont pas supportées ou non actives sur le support concerné, les règles classiques de mise en page présentes en dehors de la déclaraion *@-epubx-page-template*, doivent pouvoir s'appliquer.

Cependant si le support des spécifications est actif alors ce sont toutes les règles de mise en page qui s'appliquent : les règles établies à l'extérieur des déclaration du *@-epubx-page-template* comme celles établies à l'intérieur.

Syntaxe :

```
[classical css-rule]*

@-epubx-page-template {
    [ <css-rule> ]*
}
```

Exemple :

```
/* règle classique de mise en page de sidebar */

.sidebar {
    float: left;
    width: 20em;
```

```
}

/* règles de mise en page fluide, si l'application de lecture le
supporte, l'élément sidebar est placé dans une partition séparée
*/

@-epubx-page-template {

    .sidebar {
          float: none; /* don't float */
          width: 100%; /* use partition width */
          -epubx-flow-into:leftbox;
    }
    ...
}
```

Conclusion très personnelles

Avant la lecture de cette spécification, mon a priori était plutôt positif car elle me laissait entrevoir la possibilité de gérer des gabarits de mise en page dynamiques permettant à une mise en page fixe de posséder un placement plus adapter de ses éléments en fonction du support de lecture et de son orientation.

Cependant le premier problème que j'ai pu noter est que cette spécification s'appuie sur des spécifications CSS qui ne sont pas encore véritablement établies puisque pour la plupart elles demeurent à l'état de brouillon depuis quelques années. Ainsi on ne peut que supposer de leur devenir.

Le deuxième problème, qui m'est apparu, est l'étonnante complexité de la mise en œuvre. Bien que l'ensemble des règles ne soient définies à l'intérieur d'une déclaration globale afin de les isoler et d'éviter un quelconque souci avec les règles communes de mise en page, elles ne sont pas simples. En effet certaines déclarations ne peuvent comporter que des règles particulières, le contenu doit-être calculé au préalable etc. On aurait aimé certainement quelque chose de plus simple qui nous épargne ce genre de choses. Le principe des vases communicants reste intéressant et très logique, mais le mode de déversement des contenus est, il me semble, trop sophistiqué.

Ensuite je ne comprenais que la notion de mise en page fluide s'entendait comme une présentation différente, selon la taille d'écran et ou l'orientation, à l'intérieur d'une mise en page fixe. Le même texte était affiché différemment dans la même page écran, c'est d'ailleurs ce concept qui est pris dans *Adobe InDesign*... mais qui est complètement déconnecté de la génération EPUB. Pourtant lors de la lecture de la spécification il apparaît que le contenu se déverse de partition en partition et que la 'page suivante' est générée... Ainsi il semblerait qu'il y ait création d'une nouvelle page si le contenu initial n'est pas totalement consommé selon la règle adoptée pour un cas de mise en page. Dans ce cas, ce principe de mise en page fluide ressemblerait à un concept de mise en page de type *reflow* auquel on aurait rajouté une couche de placement ou de positionnement plus 'adapté' des éléments. Cependant les défauts connus du multicolonnage, par exemple, ne sont pas corrigés et pire, ce même comportement continue à s'appliquer. Ainsi il est impossible de

pouvoir correctement fabriquer du multicolonnage en mise en page *reflow* à la façon du *print*, et c'est bien dommage. En plus, je me pose la question de la réelle nécessité de réaliser ce type de spécification si déjà les règles des *Media Queries* peuvent réaliser certaines choses.

Enfin, étant donné que les applications de lecture se comportent de manière totalement disparate en fonction de l'orientation, je ne vois pas comment l'application de la mise en page fluide puisse se mettre en place. Peut-être que le premier pas est d'essayer de faire adopter un comportement commun ou des options de comportement communes à l'ensemble des applications de lecture. La bataille se déporterait plus du côté des fonctionnalités additionnelles entre les applications de lecture, plutôt que d'essayer d'afficher proprement un contenu, certes un peu plus complexe que du seul texte en flux, sur plusieurs supports en générant un véritable casse-tête pour être compatible et qui pour l'instant ne joue pas en faveur de la présentation des documents.