BISG
BOOK INDUSTRY STUDY GROUP

# Field Guide to Fixed Layout for Ebooks

Compiled by the Content Structure Committee of the Book Industry Study Group

Updated Version 1.2
Published October 2014

FEATURING
How to make fixed layout...
Accessibility issues...
Synching text & audio...
Retailer Standards...
Common problems...
Authoring Tools...and more!

BIC

Sne
SYNDICAT NATIONAL
DE L'ÉDITION

Endorsed by
Book Industry Communication and
The French Publishers Association

*Field Guide to Fixed Layout for Ebooks* was prepared for the Book Industry Study Group, Inc. (BISG) by members of its Content Structure Committee.

## *Revision History*

February 5, 2013: Initial Publication
June 28, 2013: Information added about Google Play Books
September 9, 2013 (Version 1.1): Added chapter on ”EPUB Fixed Layout Lessons”
October 16, 2014 (Version 1.2): Added chapter on “Authoring Tools & Limitations” and updated information in Retailer Standards chapter

ISBN 978-1-936757-18-3

Copyright © 2013, 2014 the Book Industry Study Group, Inc. All rights reserved.

The Book Industry Study Group, Inc. (“BISG”) owns all rights in the PDF of *Field Guide to Fixed Layout for Ebooks* and the copyrights therein. This PDF is made available free-of-charge. Anyone who downloads it may post a copy on his or her company’s secured intranet server or unsecured website or distribute it in physical or digital form to others.

Anyone wishing to use parts of this PDF in presentations, blog posts, speeches, etc. should first request permission of the Book Industry Study Group by emailing info@bisg.org.

Book Industry Study Group
145 West 45th Street, Suite 601
New York, NY 10036
646-336-7141
www.bisg.org

Edited by Dave Cramer, Hachette Book Group
Copyedited by Patricia Fogarty

## *Contributors:*

Laura Brady, Brady Typesetting
Garth Conboy, Google
Dave Cramer, Hachette Book Group
John Costa, S4Carlisle
Ben Dugas, Kobo Inc.
Markus Gylling, IDPF
Jean Kaplansky, Aptara
Bill Kasdorf, Apex CoVantage
Liz Kessler, Hachette Book Group
Liisa McCloy-Kelley, Penguin Random House
Julie Morris, BISG
Wendy Reid, Kobo Inc.
Jonathan Thurston, HarperCollins Publishers

# Contents

© 2014, the Book Industry Study Group, Inc.

© 2014, the Book Industry Study Group, Inc.

# Introduction

The explosive growth of ebooks and digital content in the book industry over the past several years has created myriad challenges for publishers and developers. These challenges include the fragmentation of tools, language, and formats used to create content for various reading systems, as well as a shortage of agreed-upon standards or best practices that content creators need to make smart decisions and overcome these challenges.

One hot-button issue is the application of a technique for creating static, fixed ebook “pages” called **fixed layout**. The popularity of fixed-layout ebooks has grown in some genres and dropped off in others, but many people are still unsure why and when fixed layout is a good idea.

At present, support for fixed layout from device manufacturers and retailers is inconsistent at best; there is no single standard for creating fixed-layout products and information on creating them is rapidly changing and sometimes hard to find. The Book Industry Study Group (BISG), through its Content Structure Committee’s Fixed Layout for Ebooks Working Group, has therefore created this “field guide” to fixed-layout standards, formats, and tools, to make it easier for publishers to create and sell these books.

Our intention is to update the *Field Guide to Fixed Layout for Ebooks* periodically as the implementation of fixed layout in the e-reader ecosystem changes over time.

## Intended Audience

As we’ve worked on this field guide, we’ve tried to keep two intended audiences in mind. The first are employees at a smaller publisher, who have been asked by management to create fixed-layout ebooks and are looking for a place to start. Second, we hope the document will be helpful for managers in publishing companies of all sizes looking for an overview of the processes and the possible problems surrounding the use of fixed layout.

© 2014, the Book Industry Study Group, Inc.

# Overview: Digital Formats for Fixed Layout

Let’s start off with a look at the types of digital formats for fixed layout available today.

## PDF

PDF is clearly the industry-standard print-archive format. Pixel-perfect layout and fonts may be fully retained. Desktop PDF renderers are free, ubiquitous, and generally of high quality. Device-based PDF rendering is less reliable, with issues around performance, memory utilization, lack of interactivity and accessibility, and renderer licensing. PDFs are difficult or impossible to reflow, so they generally must be letterboxed to be displayed within an available window or screen size and must be panned when zoomed. PDFs create other challenges for retailers as well. They are typically large and more difficult to transfer, they take more processor power to render and they embed font information in a way that is hard to protect with DRM. As a result, few retailers actually sell PDFs as eBooks. If they accept PDFs as input, they likely turn it into another downstream format that is easier to render and deliver.

## Image/Digital Replica

An ebook book can be created from a series of images, by essentially taking a picture of each page. This is easy to do, and can look nice, but making such books searchable and accessible requires lots of work. Various “digital replica” formats provide additional functionality on top of an image-based core, such as XML-encoded hot-spot geometry to enable navigation, text overlays to enable searching, or complementary text-only views. No standard way of doing these things has emerged.

## EPUB

Another option for producing fixed-layout ebooks is EPUB Fixed Layout (FXL) from the International Digital Publishing Forum (IDPF), which allows near pixel-perfect encoding of content, one-spine-item-per-page, utilizing predominantly HTML, SVG, or images (or a combination thereof). A common incarnation is produced using background images with absolutely positioned text-box overlays. This allows zooming/panning to great depth, with the text rendering crisply, but images eventually pixelate (unless they are SVG). The EPUB Fixed Layout format (http://idpf.org/epub/fxl/) is based on early proprietary support for “Apple Fixed Layout” in iBooks. The content markup for both (basically standard EPUB 3) is very similar; the main difference is in the metadata used to describe/specify the content as fixed layout. EPUB Fixed Layout is almost always created mechanically, often using InDesign plugins or, now, InDesign iteself.

© 2014, the Book Industry Study Group, Inc.

A related effort in process at the IDPF is EPUB Adaptive Layout, formerly known as AAL (Advanced Adaptive Layout). Eventually this is expected to provide a good balance between reflowable and fixed-layout content. The W3C CSS regions and exclusions specifications will be combined with “page templates” to support high-fidelity content that can “adapt” to multiple screen sizes, aspect ratios, and orientations. EPUB Adaptive Layout will not, however, replace fixed layout, because it will be dependent on reading systems that can execute it properly; many publishers will still find that fixed layout will be their best choice until EPUB Adaptive Layout becomes more widely implemented.

# When to Use Fixed Layout

Let’s now take a look in more detail at when fixed layout is and isn’t appropriate.

## When Is Fixed Layout Most Appropriate?

Fixed layout which exactly replicates the print design is almost always appropriate for Children’s picture books, Manga, Comics and Graphic novels. These books have art created at a particular aspect ratio, and almost always need the entire page to bleed. Retailers are moving toward consistent support for these types of books, and there has been good consumer acceptance of these genres in eBook form.

Print-replica fixed layout may be appropriate for heavily designed and illustrated titles that are more difficult to reinvent as reflowable content. In such cases, the placement of text and related tables, illustrations, sidebars, etc., is essential for maintaining the sense of the text or the story. Illustrated textbooks and full-bleed art books are a few examples of the types of books that lend themselves to fixed-layout format because of design issues. This approach may also be necessary if there is a contractual obligation to replicate the printed work.

In many cases, using fixed layout, but not replicating the print design, may be a good solution. Text can be set in a bigger font, and the design can be optimized to avoid pinching and zooming on smaller screens. This approach can work well for cookbooks, gift books, and art books

Just as for reflowable books, publishers have the option to enhance fixed-layout content with animation, interaction, narration integration, video, and sound.

However, fixed layout should not be considered the automatic default for the conversion of these types of products. Often, because we are so familiar with (and attached to) the print format, we cling to fixed layout when a reflowable digital product might be more appropriate.

## When Is Fixed Layout Not As Appropriate?

While editors and designers may be most comfortable producing an exact page replica of a print book, this isn’t always the best solution for the content, and it can limit the book’s distribution potential. Here are a few considerations to keep in mind before deciding to go with fixed layout.

A text-heavy fixed-layout title can generate an unfriendly user experience because the customer continually

© 2014, the Book Industry Study Group, Inc.

has to pinch/zoom/pan the pages on the tablet in order to view the content. There can be a loss of some of the features available in reflowable ebooks—for example, text-search functionality can be lost if the text is flattened as part of an image file. Additionally, a customer who expects a reflowable experience but encounters instead the print replica may experience confusion and disappointment.

As an alternative to fixed layout, there are creative ways of designing reflowable digital books for complex titles like cookbooks and art books. Consultation with editors and designers may be required if content needs to be reconfigured—for example, sidebars will need to be anchored to particular text, or original art may need to be redesigned to work in a single-column configuration.

From a production standpoint, fixed layout is labor-intensive and expensive. Fixed layout needs to accommodate the conversion and oversight of multiple formats—Amazon, Apple, B&N, and Kobo all support fixed-layout ebook files but use different formats, although widespread adoption of EPUB3 will help reduce this fragmentation. Each format must be individually reviewed by the publisher for quality assurance, usually multiple times, thereby creating additional work and cost for content-management departments, which in turn impact production schedules and internal resourcing. Alt-tags describing each image need to be written.

Fixed layout can also affect tight production schedules. Because the format relies heavily on the print layout, conversion efforts typically begin once the print file is finalized, even though the digital product often has to come out either at the same time, shortly thereafter, or even, in some cases, before the print edition. There is, therefore, a high risk of missing on-sale dates, which reduces the ability to capitalize on publicity and marketing.

Finally, reliance on fixed layout can hinder a publisher’s commitment to innovation at a time when experimentation is critical. While fixed layout does not, in fact, prevent the inclusion of advanced features like scripting and rich media, a focus on replicating print, coupled with the complication of creating multiple versions for multiple platforms, often inhibits the incorporation of features that truly distinguish digital from print publications.

## A Note about Fixed Layout and Distribution

The use of fixed layout can shrink distribution channels because many reading systems in the marketplace don’t support it. It’s therefore important to identify whether the goal for a given title is the widest possible distribution or simply a layout that mirrors the print work. Reflowable ebooks have the widest distribution opportunity in the marketplace.

© 2014, the Book Industry Study Group, Inc.

# Creating a Fixed-Layout Ebook

## Approaches to Fixed Layout

There are three ways to create a fixed-layout ebook:

1. Using HTML and CSS Absolute Positioning
2. Using Images
3. Using SVG

We will focus on the first method, as it has the most support in the marketplace. Image-only solutions usually result in an unsatisfactory user experience except when the viewport (the window by which the pages are viewed on the reading system) closely approximates the dimensions of print. SVG has so far not worked well for people who have tried it, due to performance issues, limited support in reading systems, and the difficulty of making corrections. This may change in the future, though, so stay tuned!

## Technical Fundamentals: The Structure of Fixed-Layout Content

The multiplicity of fixed-layout formats is a huge problem for publishers, distributors, and vendors. Ideally we would create a single version of each title, which could be used on any device and sold by any vendor.

In the meantime, most vendors agree on the fundamentals of how fixed-layout EPUB files should be created. Let’s take a look at what should work everywhere.

### *Book Content*

The primary way to create a fixed-layout title is through the use of XHTML (EPUB 2) or XHTML5 (EPUB 3), in conjunction with CSS absolute positioning. A very common situation is to have:

1. a full-page image file referenced by the HTML file, with
2. a text layer on top of the image, and
3. CSS to position each line of text exactly where it is wanted.

Let’s look at an example from a simple children’s book.

© 2014, the Book Industry Study Group, Inc.

### *HTML*

```
<?xml version="1.0" encoding="UTF-8"?>
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:epub="http://www.idpf.org/2007/ops">
  <head>
    <meta name="viewport" content="width=600, height=600"/>
    <meta charset="utf-8"/>
    <title>The Earth Book</title>
    <link href="css/stylesheet.css" type="text/css" rel="stylesheet"/>
  </head>
  <body>
    <div class="page006">
      <img src="/readermedia/rw-documents/
4715_field_guide_to_fixed_layout_for_e_books/components/006.png" alt="Girl with
card saying I love you"/>
      <p class="p6l1"><span id="s6l1">I use both sides of the paper</span></p>
    </div>
  </body>
</html>
```

Some things to note:

1. This example uses the EPUB 3 vocabulary (noticeable mostly due to the EPUB namespace).
2. `meta name="viewport"` is important! This describes the initial size of the page, and needs to match the background image, as well as the CSS.
3. Everything has an ID, which will allow us to target each individual line in the CSS.

### *Image*

Page_006.png is a 600-pixel by 600-pixel image, which matches the viewport tag above. It’s square to match the original book (and thus the original art).

### *CSS*

There’s a lot going on here, but let’s look at it piece by piece.

### *Body element*

```
body {
width: 600px;
height: 600px;
```

© 2014, the Book Industry Study Group, Inc.

```
margin: 0;
overflow: hidden;
}
```

Again, we're defining the size of the page, to match our HTML viewport.

### *Images*

```
img {
position: absolute; width: 600px; height: 600px; margin: 0;
top: 0;
left: 0;
z-index: -1;
}
```

Everything uses absolute positioning; `z-index` lets the text sit on top of the image.

### *Text*

```
p {
position: absolute;
margin: 0;
padding: 0;
font-size: 30px;
font-family: "Parr";
}
```

This defines the basic text qualities.

```
.page006 > p { top: 50px; left: 110px;}
```

Since we only have one line of text on this page, we can define the position this way. With more lines, we'd use a positioning statement for each individual line.

## Fixed Layout in EPUB 3

Fixed-layout ebooks were not officially supported in the original EPUB 3 specification. CSS absolute positioning (necessary for Apple's approach) was "discouraged," and there was no provision for the metadata needed for these types of books. An ad hoc working group was set up in late 2011 to address these issues, and a specification was approved (as an "informational document") in March 2012 (http://idpf.org/epub/fxl/). The spec standardizes the metadata needed for fixed-layout titles; in addition, it specifies how to define a fixed layout, describe the preferred orientation, control the creation of spreads, and define content dimensions. This spec does not recommend a single approach for constructing fixed-layout EPUBs; the properties apply to books that use SVG or images for content as well as the HTML + CSS absolute-positioning approach discussed above.

To use the spec, you need to add metadata to (1) the package (OPF) file and (2) the content documents themselves (usually XHTML5).

### *FXL Package Metadata*

Package metadata tells the reading system how to display fixed-layout content. There are four properties:

1. **`rendition:layout`** just specifies whether the book is fixed-layout or not.
2. **`rendition:orientation`** allows us to "lock" the orientation of a book—for example, so that it always displays in landscape orientation.
3. **`rendition:spread`** is the tricky one; it controls whether the reading system can "glue" individual pages together into a spread, as Apple's iBooks does now.
4. **`page-spread-*`** controls the placement of a single "page" when spreads are being created by the reading system. You would use this to have the first "page" appear on the right half of a spread, for example.

**Note:** Each of these properties can be used in two ways. When used in the package `meta` element, the property applies to the entire book. To do something different on an individual file (aka "spine item"), use the property on the `itemref`:

```
<itemref … properties="rendition:layout-reflowable"/>
```

This would allow you to include a reflowable section in an otherwise fixed-layout book. Unfortunately, this feature is not widely supported.

### *The Rendition Prefix*

In order to use the following properties in the package, you must declare the rendition prefix:

```
<package xmlns="http://www.idpf.org/2007/opf"
unique-identifier="bookid" version="3.0"
prefix="rendition: http://www.idpf.org/vocab/rendition/#">
```

### *The `rendition:layout` Property*

If a book is in fixed layout, then this property should be set to `pre-paginated`:

```
<meta property="rendition:layout">pre-paginated</meta>
```

If a book is a regular, reflowable ebook, you can omit this or use the default:

```
<meta property="rendition:layout">reflowable</meta>
```

### *The `rendition:orientation` Property*

To “lock” the orientation of a device to landscape:

```
<meta property="rendition:orientation">landscape</meta>
```

To “lock” the orientation to portrait:

```
<meta property="rendition:orientation">portrait</meta>
```

If either orientation works, you can omit this or use the default:

```
<meta property="rendition:orientation">auto</meta>
```

**Note:** The property will have meaning only when the reading device can respond to a change in its orientation, as with an iPad.

### *The `rendition:spread` Property*

The current leading implementations present a dilemma in their contradictory approach to spreads. Apple’s iBooks does an interesting thing: it takes two separate content files and displays them side by side in landscape orientation. Essentially, iBooks asks you to create individual pages (one page per HTML file) and then builds spreads out of those files. Amazon’s basic KF8 format (which is not EPUB 3) asks you to create a spread in each file and then just displays that single file on-screen.

**Note:** The best approach as of this writing is to create one HTML file per spread, and set `rendition:spread` to `none`.

So how do we tell the reading system when it's OK to build these "synthetic spreads?" This is the purpose of the `rendition:spread` property. There are five possible values for this property:

1. **`none`** means that the reading system will never try to create a spread out of individual content files.
2. **`auto`** lets the reading system decide.
3. **`both`** tells the reading system to always assemble content documents into spreads, regardless of the device's orientation.
4. **`landscape`** means that the reading system should create spreads only when the device is in landscape orientation.
5. **`portrait`** means that the reading system should create spreads only when the device is in portrait orientation.

A few examples will help to make sense of this. Probably the most common situation today is for books based on Apple's guidelines, with one content file per page, designed to be displayed in spreads when in landscape orientation. You could say:

```
<meta property="rendition:spread">landscape</meta>
<meta property="rendition:orientation">landscape</meta>
```

This locks the device to the landscape orientation and tells it to make spreads out of the pages.

If, on the other hand, your content files are each one spread, as Kindle's KF8 wants, then you need to tell other reading systems to not create spreads:

```
<meta property="rendition:spread">none</meta>
```

### *The `page-spread-*` Properties*

If your reading system is creating spreads and you have a special page (like a cover) that isn't joined up with another page as a spread, you can control whether it sits on the left half of the spread (`page-spread-left`), the right half (`page-spread-right`), or right in the middle (`page-spread-center`).

### *FXL Content Metadata*

Content metadata has one purpose: to define the initial dimensions of the content. For HTML content, this is done using the viewport `meta` tag in HTML:

```
<head>
  …
  <meta name="viewport" content="width=1200, height=600"/>
  …
</head>
```

© 2014, the Book Industry Study Group, Inc.

For SVG content, the size is expressed using the `viewBox` attribute. For image content, the pixel dimensions of the image define the size, so additional metadata is not required.

### *Non-standard Metadata*

The `com.apple.ibooks.display-options.xml` file (found in the META-INF directory within the EPUB package) is valid in EPUB 3, but ideally we won't need to use it. Similarly, a valid EPUB 3 can contain other metadata, but we encourage people to use the IDPF metadata whenever possible.

© 2014, the Book Industry Study Group, Inc.

# Accessibility

## Who, What, When?

When approaching the topic of accessibility in the context of ebook design in general and fixed layout in particular, it is important to keep in mind that accessibility is a multi-faceted problem area. As discussed in depth in the O'Reilly publication *Accessible EPUB 3* (available as an ebook, free of charge), the user groups affected are many, and while the reading needs of these groups vary, they can typically be met collectively by a properly designed ebook, where "properly designed" is a collection of dos and don'ts outlined in the *Accessible EPUB 3* publication.

Who, then, are the users who may potentially find your ebook inaccessible? You may have heard the term *print disability*, typically defined as any condition in which a user is unable to read or use standard printed material due to blindness, visual disability, physical limitations, organic dysfunction, or dyslexia.

Within the dyslexia research field, it is generally agreed that 1 out of 10 people has some form of dyslexia. Similarly, research from the European Union has shown that 21% of people over the age of 50 experience severe vision, hearing, or dexterity problems. So we are by no means talking about small niche populations, and as international demographics shift toward increasingly aged populations, the numbers only get bigger.

Another category of users, relevant to the topic at hand, are those who are subject to a *situational disability*, defined as the *temporary inability to interact with the content in the reader's preferred modality*. Anyone can become situationally disabled at any time. Examples include:

- Inability to view a screen while outdoors because of screen glare
- Inability to efficiently consume content due to screen limitations (e.g., the small screens of cell phones)
- Inability to hear sound while in a noisy environment, such as an airplane, bus, or subway

## Risk Factors in FXL

What does all of this have to do with fixed-layout content? Fixed-layout content runs the risk of possessing some or all of the properties that make hard copy print inaccessible to the aforementioned user groups. Following are brief descriptions of the core risk factors for fixed layout.

© 2014, the Book Industry Study Group, Inc.

### *Fixed-layout content cannot adapt to devices, users, and contexts*

Fixed-layout publications are by nature restricted in the ways that they can adapt to users and usage contexts. While the publisher (as discussed in the "When Is Fixed Layout Most Appropriate?" section above) has deliberately made the choice to supply the content using fixed layout, the negative impact of this choice on accessibility (and general usability) can be quite dramatic:

- A user with dyslexia, low vision, or color blindness who depends on special color schemes and/or font settings in order to consume content effectively will find that setting these user preferences either is not possible in the reading system or causes side effects when activated that make the content illegible. The same problem may occur in the context of "night and day modes" provided by reading systems.
- A user trying to use a cell phone to read fixed-layout content designed for a tablet will report significant decrease of efficiency and efficacy of information retrieval due to the need to resort to extensive panning and zooming (see the discussion of *situational disability* at the end of the "Who, What, When" section, above).
- A user with dyslexia or sight loss who depends on the audio modality by use of a screen reader may find this feature to be unavailable or highly unreliable (see next entry).

### *Some fixed-layout content cannot be accessed by assistive technologies*

Assistive technologies such as screen readers and Braille displays need access to the publication text in marked-up HTML or SVG form in order to transpose the content to the audio (in the case of screen readers) and touch (in the case of Braille displays) modalities. When the fixed-layout content is designed such that the publication text is embedded in images and not as a separate HTML or SVG layer of text, the entire publication becomes inaccessible to users of such devices. (Another consequence of completely image-based publications like these is that the content cannot be searched or indexed—unless the reading system has a built-in OCR feature.)

### *Some fixed-layout content loses the logical reading order*

When CSS absolute positioning is used in XHTML- and SVG-based fixed-layout documents, it may contradict the logical reading order of the underlying document itself; in other words, the order in which the content elements appear in the document markup may not match the order in which they appear on-screen. For assistive technologies such as screen readers and Braille displays (which access the structure of the underlying document), such out-of-sync content poses a severe risk of making the document inconsistent or confusing at best, and illegible at worst.

## Upcoming Solutions

In the context of EPUB 3, we need to be aware that the IDPF is pursuing a number of projects that, once mature, will significantly enhance the publishers' toolkit by enabling the design and provision of content that can better adapt to devices, users, and contexts, all while retaining the desired rich-layout characteristics.

- **Adaptive Layout (AL)** defines a model for template-based paginated layouts, allowing content to be formatted into a sequence of richly designed interactive pages, rather than being presented as a single scrolled container or as a scrolled container that has been simply split up into a sequence of pages (aka classic dynamic pagination). Once Adaptive Layout is supported by authoring tools and reading systems (at the time of writing, Adaptive Layout authoring is supported by InDesign CS6), it will allow publishers to provide content that can adapt intelligently to different devices, while still allowing for the layout paradigm and user experience typically associated with fixed-layout content.
- **Advanced Hybrid Layout (AHL)** is a project that, among other things, aims to define how to embed multiple renditions of the same publication into one EPUB file, and how to allow for selection and mapping between these renditions. As an example, AHL will allow an image-based fixed-layout publication to have an HTML text-based "sibling" that can be selected by the user as the preferred reading mode at any time (e.g., the Barnes & Noble "article mode") or serve as the data source for assistive technologies, such as screen readers or Braille displays. A publication that contains multiple renditions in this way can fulfill accessibility and usability criteria, as well as lending itself perfectly well to bread-and-butter functionality, such as searching and indexing.

Until the above extensions to EPUB are available for use in the real world, however, publishers who use fixed-layout content need to choose an approach that runs the least risk of negative effects on the user. The next section outlines a recommended authoring strategy to meet this goal.

## Recommended Authoring Approach

In order to maximize the accessibility and usability of fixed-layout content, the following authoring approach is recommended:

1. Produce reflowable documents instead of fixed-layout documents if you can.
2. When producing fixed-layout documents, don't use image-only documents; instead, use XHTML or SVG with embedded text.
3. Use the XHTML or SVG document structure and CSS absolute positioning in a way that keeps the logical reading order in sync with respect to the document element order and the order in which those elements appear on-screen.
4. Avoid layering content in ways that is difficult to perceive due to small font size or low color contrast.
5. Consider using JavaScript pop-ups to enlarge text when it is particularly small.

For recommendations for interactive/scripted content, refer to the section on interactivity and ARIA in the O'Reilly publication ***Accessible EPUB 3*** (note that these apply equally to both reflowable and fixed-layout content).

Similarly, keep in mind that the Web Content Accessibility Guidelines (WCAG) apply to EPUB content, both reflowable and fixed-layout.

# Synching Text and Audio with Media Overlays

A major new feature of EPUB 3 is the ability to link an audio recording to text and to keep the two synchronized. Elements at the word, sentence, paragraph, and page level can be highlighted as the corresponding audio is heard. Various retailers call this feature “Read and Listen” or “Read to Me,” as it’s been widely used for fixed-layout children’s picture books. Currently, some reading systems support this feature only for fixed layout, even though the concept should apply to all books.

EPUB 3 allows this synchronization through media overlays, which is a subset of the pre-existing SMIL (synchronized multimedia integration language) standard. Essentially, an XML file (using the SMIL vocabulary) contains a pointer to each text element in the document and pairs it with the start and end times of the audio file that refers to that text:

```
<par id="para1">
  <text src="chapter_002.xhtml#c002p0001"/>
  <audio src="mobydick.mp3" clipBegin="0:14:48" clipEnd="0:15:13"/>
</par>
```

Note that the `par` element is grouping a text reference (to a particular paragraph in chapter 2) with an audio reference (the 25 seconds of sound beginning at 14:48 in a particular MP3 file). In this case, we’re synchronizing only at the paragraph level, but it’s possible to point to every single word in the text. In that case, the audio segment you’re identifying may be less than one second long.

The difficulty is in creating this SMIL file. You need to know exactly when the audio for each element begins and ends. It’s possible to do this by hand, but any realistic production workflow would need an automated solution to generating this information. And of course you need XHTML; this can't be done with image-based fixed-layout publications.

## The SMIL File

Here’s an example of a single SMIL file, which corresponds to a single page of a children’s picture book:

```
<?xml version="1.0" encoding="UTF-8"?>
<smil xmlns="http://www.w3.org/ns/SMIL"
      xmlns:epub="http://www.idpf.org/2007/ops" version="3.0">
  <body>
    <par id="id-008">
```

```
      <text src="/readermedia/rw-documents/
4715_field_guide_to_fixed_layout_for_e_books/page_009.html#span9line1"/>
      <audio clipBegin="0:00:00.000" clipEnd="0:00:01.0"
              src="/readermedia/rw-documents/
4715_field_guide_to_fixed_layout_for_e_books/audio/page009.m4a"/>
    </par>
    <par id="id-009">
      <text src="/readermedia/rw-documents/
4715_field_guide_to_fixed_layout_for_e_books/page_009.html#span9line2"/>
      <audio clipBegin="0:00:01.0" clipEnd="0:00:02.4"
             src="/readermedia/rw-documents/
4715_field_guide_to_fixed_layout_for_e_books/audio/page009.m4a"/>
    </par>
  </body>
</smil>
```

In this case, each page has a separate audio file, so line 1 of the text is heard from 0.0 seconds to 1.0 seconds, and line 2 is heard from 1.0 seconds to 2.4 seconds. The corresponding content file looks like this:

## The Content File

```
<?xml version="1.0" encoding="UTF-8"?>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:epub="http://www.idpf.org/2007/ops">
  <head>
    <meta name="viewport" content="width=600, height=600"/>
    <meta charset="utf-8"/>
    <title>The Earth Book</title>
    <link href="css/stylesheet.css" type="text/css" rel="stylesheet"/>
  </head>
  <body>
    <div class="page009">
      <img src="/readermedia/rw-documents/
4715_field_guide_to_fixed_layout_for_e_books/components/009.png" alt="Owl family in
tree"/>
      <p class="page9line1"><span id="span9line1">and I want the owls to
have</span></p>
      <p class="page9line2"><span id="span9line2">a place to live</span></p>
    </div>
  </body>
</html>
```

© 2014, the Book Industry Study Group, Inc.

Notice how the `text` element in the SMIL file points to a line in the content file, and the `audio` element points to the corresponding audio.

## The Package File: Media Overlay Metadata

Media overlays require some special coding in the EPUB 3 package file. First, you must associate the SMIL file with the corresponding XHTML content file:

```
<item id="pg009" href="page_009.html" media-type="application/xhtml+xml"
media-overlay="page_009-overlay"/>
```

Second, you must list the SMIL file itself:

```
<item id="page_009-overlay" href="smil/page_009.smil" media-type="application/
smil+xml"/>
```

Note how the `media-overlay` attribute in the content item points to the ID of the SMIL file. Finally, there is metadata for the media overlays themselves:

```
<meta property="media:duration" refines="#page_009-overlay">0:00:02.4</meta>
<meta property="media:duration">0:23:46</meta>
```

You need to list the duration of each SMIL file, as well as the total for all SMIL files.

## Highlighting Text: Media Overlays and CSS

Finally, we need to highlight the text that corresponds to the audio. This is done in the CSS:

```
.-epub-media-overlay-active {
background-color: #abc;
}
```

Note that you can use any class name for this property, but you must declare the name in the package file:

```
<meta property="media:active-class">-epub-media-overlay-active</meta>
```

That’s a lot of moving pieces!

© 2014, the Book Industry Study Group, Inc.

# Interactivity and JavaScript

EPUB 3 allows (but does not necessarily encourage) the use of scripting to add interactivity and other features to ebooks. This is likely the most contentious and complex part of ebooks today. Inconsistent, nonexistent, or deliberately limited support from reading systems means that it's impossible to generalize about this topic. Apple's iBooks platform allows publishers the most latitude, and has also created some Apple-specific JavaScript libraries to enable certain types of effects. Amazon's KF8 does not allow any scripting at this time.

Those interested in creating interactive fixed-layout ebooks at this time should refer to Apple's documentation, available via iTunes Connect.

© 2014, the Book Industry Study Group, Inc.

# Authoring Tools & Limitations

There are now existing software applications that allow creation of fixed-layout ebooks, which are attractive to publishers because they are fairly simple to use, fit easily within existing workflows for book design, and require little knowledge of XHTML or CSS to use. While these applications can be useful for ebook production for these reasons, some special considerations need to be taken into account when using them to create fixed-layout files.

As with reflowable ebooks, a robust QA process is a necessity, and files output from tools may need further modifications.

## Adobe InDesign

Adobe released a new fixed-layout EPUB export option for InDesign CC, in June, 2014, and updated it in October. This option exports near-exact EPUB replicas of the print design file, but it does have some drawbacks.

Many of the issues described below may be fixed in subsequent updates to InDesign CC.

### *Accessibility and Document Structure*

The most significant problems can compromise accessibility or document structure.

- Because every single word is marked up with a span and a character style override, text selection doesn’t behave as expected, and searchability is hampered.
- The export renders every item on the page in the order in which it was added—which is not necessarily the same as the reading order.
- Semantic structure is completely absent—lists, headings, and tables are not rendered as such in the EPUB.
- Semantic inflection using the EPUB3 vocabulary is minimal.
- Text set on a curved path may be deleted from the output.

### *Compatibility*

- Converting InDesign fixed layout to Amazon KF8 fixed layout might be difficult. InDesign does not create region magnification, and there may be issues with font encryption.
- InDesign does wrap every word in a span, which is necessary to create media overlays. But the dense

© 2014, the Book Industry Study Group, Inc.

markup may make it difficult to apply necessary tagging.

### *Further Reading*

For an in-depth review of InDesign’s fixed-layout EPUB export feature, see Laura Brady’s blog post at http://epubsecrets.com/adobes-new-fxl-export.php.

For more on accessibility issues, see Matt Garrish’s smart write-up at: http://matt.garrish.ca/2014/06/accessible-indesign-fixed-layouts/.

# Retailer Standards

Here we will look at the considerations needed when creating fixed-layout ebooks for various retail platforms, including Amazon Kindle; Apple iBooks; Google Play Books, Barnes & Noble Nook, and Kobo.

## Amazon Kindle Format 8 (KF8)

KF8, Amazon's proprietary format for the Kindle, was announced in late September 2011 and introduced in October. It appears to be very similar to EPUB 3, supporting fixed layout using HTML 5 and CSS 3 absolute positioning, but using different metadata than Apple or the IDPF FXL spec. (Note that the IDPF EPUB 3 FXL metadata specification provides a "concordance" between IDPF metadata and KF8 metadata for fixed layout.) KF8 fixed layout currently supports only children's books and comics/manga/graphic novels.

Adding the following five configuration metadata values to the package file defines all page and layout parameters.

1. **Fixed Layout**, which is a true or false setting.
2. **Original Resolution**, which tells the reader the aspect ratio that the content is originally designed for. This needs to be in pixels (e.g., 1024 x 800). This ratio dictates how the Kindle Fire will scale the pages. If the device aspect ratio is not matched, the Kindle Fire will resize the content to fit and insert white space around the page.
3. **Orientation Lock** can be `portrait`, `landscape`, or `none`. `portrait` locks the content to the portrait orientation, and `landscape` locks it to the landscape orientation. `none` (the default) allows either choice (but may not be supported for all devices). This parameter is required for children's books, and is optional for comics.
4. **Book Type**, which has two default values based on the original intent of fixed layout. The values can only be set to `children` or `comic`.
5. **Region Magnification**, which is a `true` or `false` setting telling the reader whether to allow magnification or not.

Orientation is a required parameter, which means that you cannot have a book that can be viewed in both portrait and landscape modes. Your designer will need to choose one of the options.

KF8 fixed layout generally supports one HTML page at a time. Designers must create a single (HTML) page for pages in portrait mode or a single HTML page containing both pages of the two-page spread for landscape mode. This restriction is relaxed for comics/manga, where it's possible to use `page-spread-*` properties to control spread creation.

© 2014, the Book Industry Study Group, Inc.

As will all devices, the screen size and aspect ratio may not match pre-existing designs. Content may be scaled to fit the screen, centered, and surrounded by white space.

The KF8 fixed-layout format lacks pan and zoom. Only region magnification is supported for enlarging text view.

KF8 features a built-in Panel View that allows comic books and graphic novels to be presented in high resolution color.

Similar to EPUB 3, KF8 fixed layout supports embedded fonts. They need to be either TrueType or OpenType. Type 1 fonts are not supported.

## Apple

Apple introduced a fixed-layout ebook format in December 2010. Originally based on EPUB 2 and now accommodating EPUB 3, it uses HTML with CSS absolute positioning for content, and a nonstandard metadata file. Apple supports most of the EPUB3 FXL spec, but there are some omissions.

### *Basic Content*

Apple uses HTML with CSS absolute positioning, as described above. For older versions of iBooks, each HTML file represented a single page of the book, and iBooks assembled those pages into spreads automatically. Now, with proper metadata, one HTML file per spread is supported (and aligns with the requirements for KF8).

### *Images*

Apple had a longstanding limit of 2 million pixels per image, but this has been increased to 3.2 million pixels for interior images.

### *Metadata for EPUB2*

Apple uses a proprietary approach for metadata for EPUB2 Fixed Layout titles. Create a file named `com.apple.ibooks.display-options.xml`, which lives inside the META-INF directory of the EPUB. Here’s the simplest example:

```
<?xml version="1.0" encoding="UTF-8"?>
<display_options>
  <platform name="*">
    <option name="fixed-layout">true</option>
  </platform>
</display_options>
```

This indicates that the EPUB2 is indeed in fixed layout.

### *Apple Display Options for EPUB2*

The chart below documents the display options offered by Apple’s fixed-layout format.

| Option Name | Possible Values | Description |
|---|---|---|
| `platform` | * | Title will be available on both iPhone and iPad. |
| | `undefined` | Available only on iPhone (and iPod Touch). |
| | `ipad` | Available only on iPad. |
| `fixed-layout` | `true` | Book is in fixed layout. |
| | `false` (default) | Book is reflowable/flowing. |
| `orientation-lock` | `landscape-only` | Book will display only in landscape orientation. |
| | `portrait-only` | Book will display only in portrait orientation. |
| `open-to-spread` | `true` | Book will open to a two-page spread. |
| | `false` | The starting page (not spread) will be enlarged to fit the viewport. |
| `specified-fonts` | `true` | iBooks will not override CSS font assignments. |
| | `false` | Under some circumstances, document font assignments will be overridden. |

### *Metadata for EPUB3*

Apple now uses the standard IDPF FXL package metadata for EPUB3 files. A few Apple-specific properties are also defined, most of which function as described above, just with a different syntax:

```
<meta property="ibooks:ipad-orientation-lock">portrait-only</meta>
<meta property="ibooks:ipad-orientation-lock">landscape-only</meta>
<meta property="ibooks:iphone-orientation-lock">portrait-only</meta>
<meta property="ibooks:iphone-orientation-lock">landscape-only</meta>
<meta property="ibooks:specified-fonts">true</meta>
```

The following property will hide the binding in two-page spreads. The default is `true`.

```
<meta property="ibooks:binding">false</meta>
```

© 2014, the Book Industry Study Group, Inc.

### *Audio Sync*

Apple has implemented the EPUB 3 standard for synced audio and text using SMIL.

### *Interactivity*

Apple supports JavaScript for interactivity in ebooks. There is limited support for local storage for variables, but content authors should check Apple's documentation for details.

### *For more information…*

Apple's documentation is quite good, but it is available only to those with an iTunes Connect account.

## Google Play Books

Google Play Books supports EPUB Fixed Layout content identified by either the older "Apple Fixed Layout" metadata (above) or with the standard EPUB 3 package-based fixed-layout metadata.

### *Basic Content*

Google Play Books supports HTML with CSS absolute positioning. Bitmap images or SVGs may be referenced by XHTML spine items to create simple image-only fixed-layout pages. Each XHTML file must represent a single page of the book and must contain a `meta name="viewport"` within the `head` element. Spreads will be assembled automatically.

### *Audio Sync*

Google Play Books supports EPUB 3 standard "media overlays" for synced audio and text using SMIL. If "read along" highlighting is desired, the styling to use must be specified using the `media-overlay-active` or `-epub-media-overlay-active` CSS class name (or an alternate class name specified by `<meta property="media:active-class">`*`class-name`*`</meta>` metadata).

### *Restrictions*

JavaScript interactivity is not yet supported (books using it are accepted, but the JS is ignored). These restrictions are expected to be lifted in future versions of Google Play Books.

## Barnes & Noble

Barnes & Noble does not currently support EPUB3 FXL, but does have a number of proprietary fixed-layout formats. Note that not every format is supported by every device.

### *epib*

This is a proprietary NOOK Kids fixed-layout format packaged as epub. The epib format offers audio, page zoom, text zoom, and custom animations. Navigation is by page turn or thumbnail bar.

### *epdf (also known as PagePerfect)*

This is a NOOK fixed-layout format using PDF. The format offers page fidelity in either portrait or landscape view. Navigation is by page turn, linked contents page, drop-down TOC, or thumbnail bar. Features include page zoom, searchable text, internal and external linking, highlighting and notes.

### *drp*

This is a NOOK proprietary graphic novel format packaged as epub. The format offers page fidelity in either portrait or landscape view. Features include page zoom, panel-by-panel navigation, and right-to-left navigation for manga.

## Kobo

Kobo has five reading platforms (eInk, Desktop, iOS, Android, and Windows 8), and supports both EPUB2 and EPUB3 fixed-layout formats. Kobo supports most of the EPUB3 FXL spec, but recommends that content creators thoroughly test their content on its platforms. EPUB2 fixed-layout titles are distributed only to iOS and Android. EPUB3 fixed-layout titles are distributed to all platforms.

### *Basic Format*

Kobo uses HTML with CSS absolute positioning, and recommends that content creators avoid the use of inline styling for placement purposes. Using inline styling can cause errors on the eInk, Android, and Windows 8 platforms.

### *Images*

Kobo supports the core image types outlined by the IDPF, and advises a maximum image size of 3MB for optimal reader experience. Kobo also recommends that image widths in CSS be declared in pixels, not percentages. For cover images, Kobo recommends a 3:4 (width:height) ratio, in any size.

### *Metadata for EPUB2*

Kobo uses a similar approach to Apple for metadata for EPUB2 fixed-layout titles. A file named `com.kobobooks.display-options.xml` is used, which resides in the META-INF folder of the EPUB. Here is an example:

```
<?xml version="1.0" encoding="UTF-8"?>
<display_options>
  <platform name="*">
    <option name="fixed-layout">true</option>
  </platform>
</display_options>
```

Kobo will also accept the `com.apple.ibooks.display-options.xml` file, but recommends that content creators use the Kobo file type for optimal results.

### *Metadata for EPUB3*

Kobo uses the standard IDPF FXL metadata for EPUB3 files. Kobo supports all five `rendition:spread` properties. It is important to note that `rendition:spread` properties are only read at the book level. Kobo intends to expand this support in future versions of its reading platforms.

### *Audio/Video Support*

Kobo supports Embedded Audio/Video as outlined in the EPUB3 spec through HTML5 on its tablet platforms (iOS and Android). Kobo also supports SMIL-synced audio and text.

### *Interactivity*

Kobo supports Javascript on its tablet platforms (iOS and Android), but recommends that content creators test thoroughly to ensure interactivity is working. Kobo supports fallback statements on its other devices.

### *Image Only Fixed-Layout*

Kobo uses a special rendering mode for EPUBs that use only images, called the Comics Renderer or Image-Only Renderer. This rendering mode allows fixed-layout EPUB files that use only images in the HTML documents (no text or links) to load faster and creates a slideshow-like reading experience. This is optimized for comics and other image-heavy documents that require faster page turns.

### *For more information…*

Kobo has published an EPUB spec available on GitHub (http://github.com/kobolabs/epub-spec), and encourages content creators to contact Kobo with any questions or issues at renderingissues@kobo.com.

# EPUB Fixed Layout Lessons

Publishers should adopt conventions that ensure their books can be displayed with full fidelity on all devices. To this end, the following pages provide insights regarding how to develop cross-platform EPUB Fixed Layout content. Along with the desire to support standards, following these guidelines provides a means of ensuring that publishers efficiently create a consistently good experience across all platforms and Reading Systems.

The pages below describe various EPUB Fixed Layout issues that have been encountered in real-world content, the root cause of the issue, and changes that may be made to ameliorate the problems.

## Platform-specific Font Assumptions

Many early EPUB Fixed Layout titles were authored for the Apple iBooks platform and were based on the incorrect assumption that the iOS font set would be available across all reading systems and rendering engines. iOS uses Times New Roman for Serif and also includes a large collection of other specific fonts (e.g. Helvetica, Arial); however, this same set of fonts is not available on all other platforms. If your design depends on a specific font, that font *must* be embedded in the EPUB file.

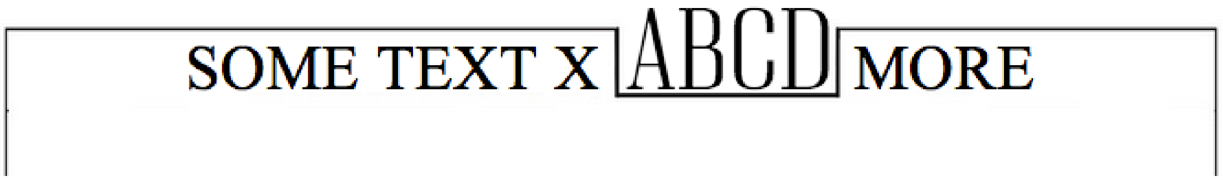
Further, even with all fonts embedded, there will still be some rendering/alignment differences between platforms. There will be slight differences (that will result in a pixel or two of rendering placement variance) caused by differing versions or implementations of kerning, adherence to font metrics, grid-hinting, LCD smoothing, and the details of justification. The following sections provide some hints for avoiding these problems that are really inherent in web-engine-based FXL rendering.

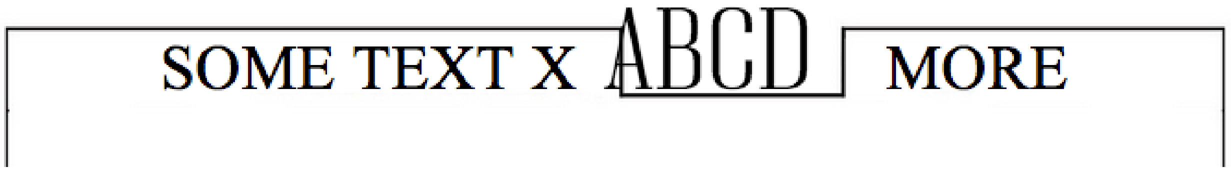
**Reading System-Specific Note:** The Google Play Books Android client provides “Tinos” as the default serif font and will map requests for “Times New Roman” to “Tinos.” Tinos has metrics that should be identical to Times New Roman, so desired results, pending glyph availability, should be obtained. All other fonts must be embedded.

## Include All Glyphs

Just embedding the desired fonts is sometimes not sufficient. Care must be taken to ensure that all glyphs expected to the pulled from the embedded fonts are actually present in the fonts. Note the following two images:

© 2014, the Book Industry Study Group, Inc.





The first one is clearly the desired version, with the text “correctly” layered into the rules provided by the background image. The markup for the header is as follows:

```
<div class="page_item1">
<p class="top"><span class="classy">SOME TEXT X&#x2009;&#x2009;</span><span
class="classyToo">ABCD&#x2009;</span><span class="classy">MORE</span></p>
</div>
```

With relevant styles being:

```
div.page_item1{
        position: absolute;
        top: 43px;
        left: 23px;
        right: 0px;
}

p.top{
        font-size: 27px;
        line-height: 23px;
        text-align: center;
        text-indent: 0px;
        margin-top: 0px;
        margin-bottom: 0px;
}

span.classy
{
color:black;
font-family: "Tinos";
}
```

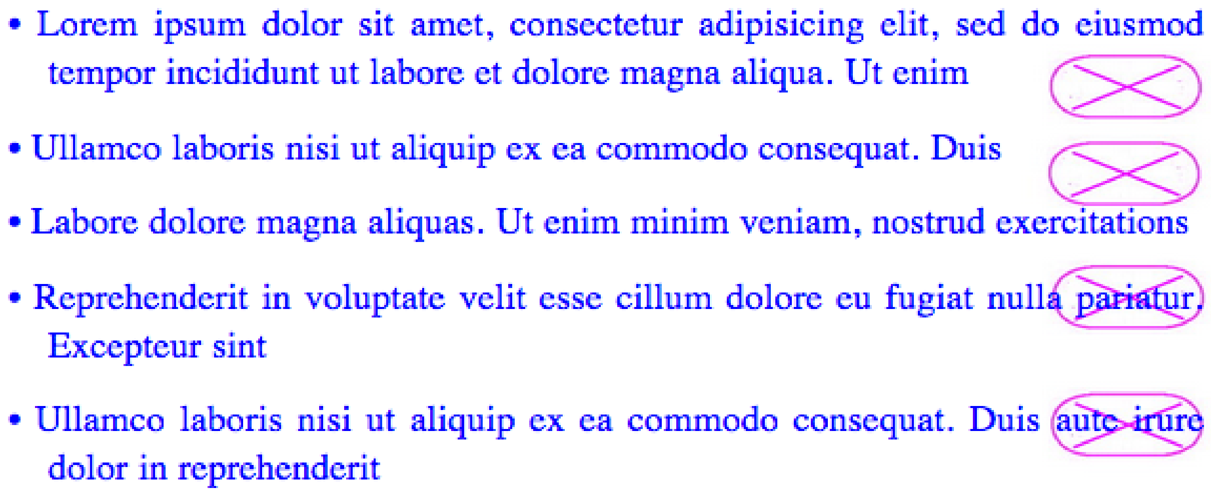
```
span.classyToo
{
letter-spacing:1px;
font-size: 51px;
font-family: "GiorgioS-Regular";
}
```

The referenced fonts are all embedded in the EPUB. The “&#2009;” is a Unicode “thin space”: this glyph is expected to be pulled from the two referenced fonts by the above markup and styling. However the glyph was only present in one of the two fonts (Tinos, but not Giorgio), causing a browser-specific fallback to be used for the missing glyph, which in some cases yields a “box” character, and in others a very large space, and in others a desirably thin space. This behavior will vary from platform to platform and can not be depended upon. Embedded fonts must contain all glyphs that are to be pulled from the fonts. Alternately, in this case, absolutely positioning the three spans of text would obviate the glyph presence issue.

A particularly common problem is lack of inclusion of “em space” and “en space” glyphs in embedded fonts. These characters are often used for positioning, and if they are missing from the desired fonts, the platform-specific backstop choice may have wildly different metrics yielding very undesirable results.

## Be Wary of Text Justification

Using `text-align: justify` especially over long flows of text, is likely to cause problems. See the following image and note the overlap of text with the bottom “oval X”s.



- Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim
- Ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis
- Labore dolore magna aliquas. Ut enim minim veniam, nostrud exercitations
- Reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint
- Ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit

The intent is to fit text in with all of the “oval X” images (which are really part of a single background image). However, the markup is requesting the text to `justify`, which lets the browser/WebView determine line breaks and also compress spaces to avoid line breaks. Specifically, it is expected that “exercitations” at

© 2014, the Book Industry Study Group, Inc.

the end of bullet #3 will break to the next line, in which case everything will line up acceptably. However, some browsers/WebViews compress the spaces on that line to make the text fit in one line; thus, the text and images also overlap in all the subsequent bullets.

This is an inherent issue with fixed layout – giving the browser the authority to break lines may cause problems, especially when trying to end lines of text at exact locations, bordered closely by images. This may, in many cases, be resolved by either hand-breaking the lines, or by absolutely positioning each line (or words within the line). Absolutely positioning lines, words, or even characters can likely solve platform-differences in “justify” implementations at the cost of slightly ragged-right appearances. One should also be aware of specifying a “box” that is insufficiently wide for all of the desired content, causing undesired wrapping (see below); leaving a few pixels “slop” on the right is not bad practice.

In some cases, more granular positioning of images within the text flow could helpful be. In the above example, absolutely positioning each question could be a good solution, or you might consider floating the “oval X”s as images to the right within boxes containing the text.

## Beware of Overly Constrained Boxes

In the following zoomed image of a “table of illustrations” note the “DES” overlapped with “ILLUSTRATIONS”:

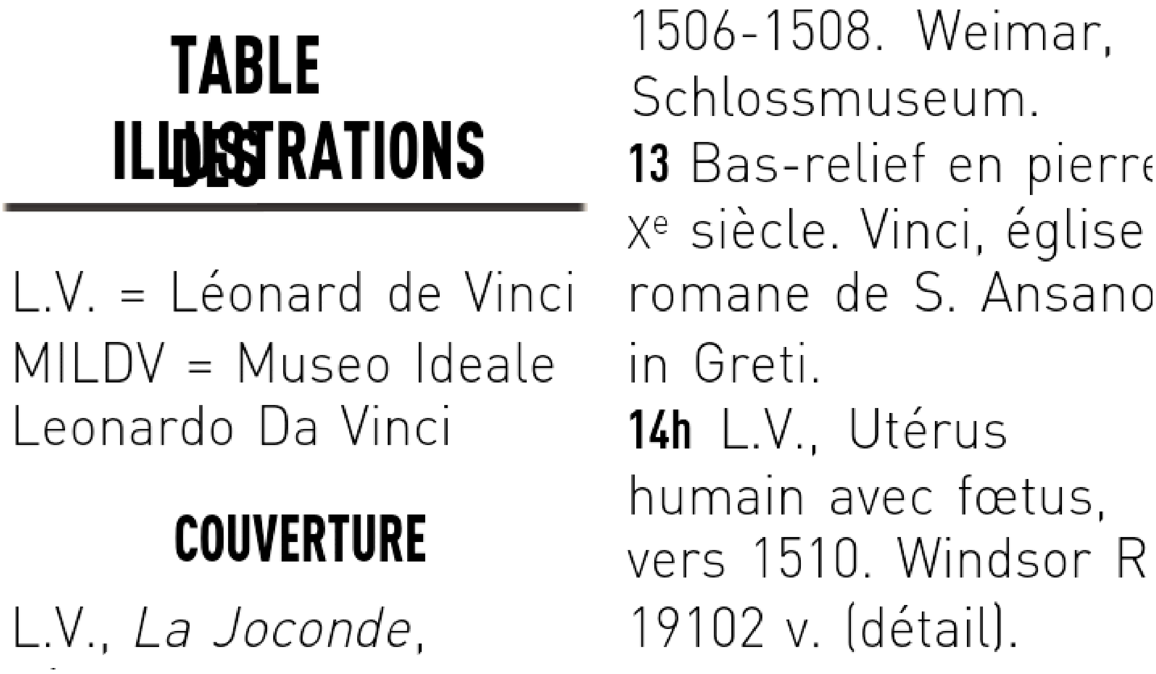


**TABLE**
**ILLUSTRATIONS**
**DES**

L.V. = Léonard de Vinci
MILDV = Museo Ideale
Leonardo Da Vinci

**COUVERTURE**

L.V., *La Joconde*,

1506-1508. Weimar,
Schlossmuseum.
**13** Bas-relief en pierr
X$^{e}$ siècle. Vinci, église
romane de S. Ansano
in Greti.
**14h** L.V., Utérus
humain avec fœtus,
vers 1510. Windsor R
19102 v. (détail).

The table header:

TABLE DES

ILLUSTRATIONS

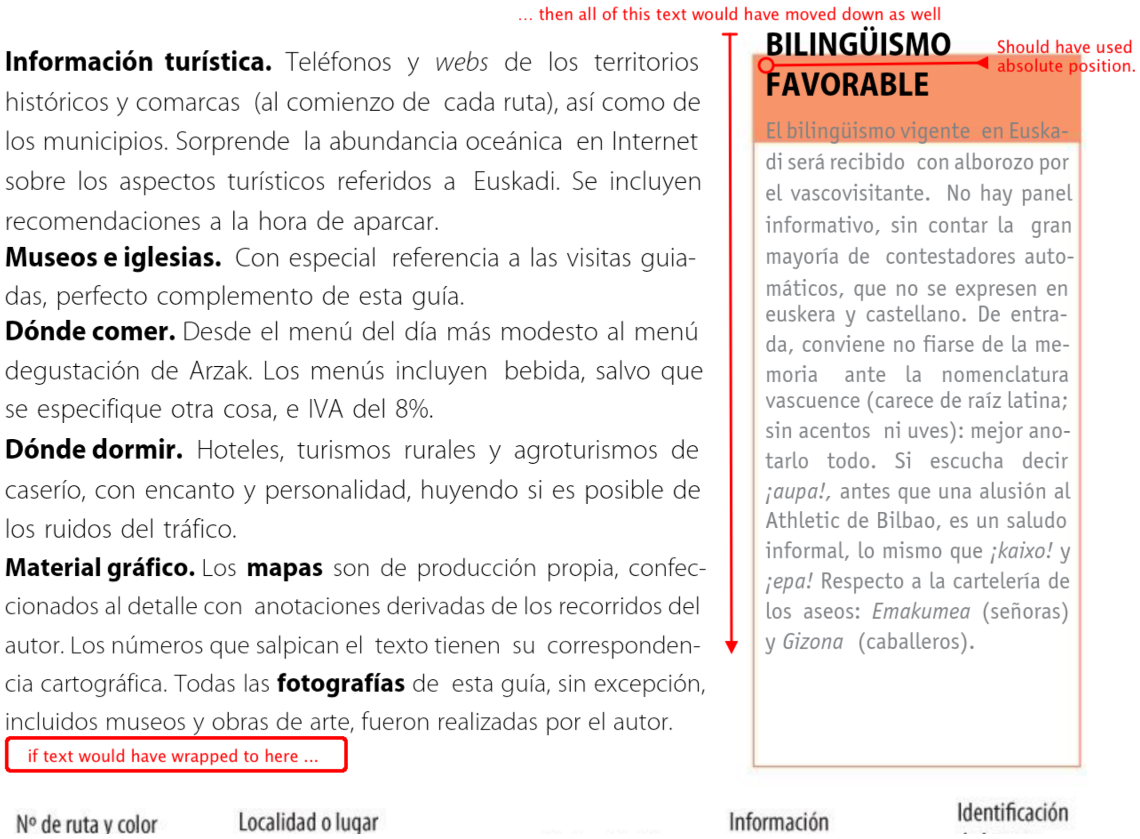
Is marked up as follows:

```
<div class="li c2 c39 c193 c12 c13 c6F" style="top:98px;left:155px;width:79px;">
  <b class="f11 c12 c194"><a id="pg_132"/> TABLE DES</b><span class="j">
 </span></div>
<div class="li nw c2 c39 c193 c12 c13 cC4 c14 f11" style="top:120px;left:140px;">
  <b class="c12 c194"> ILLUSTRATIONS</b></div>
```

The issue is that with a width of `79px` in the first `div`, the “DES” text on the first line wraps to a 2nd line, and then the “ILLUSTRATIONS” text overlaps it. This markup file renders with the undesired overlap on both Chrome and Safari on the desktop (but presumably not on iOS). In this case there is no real need to constrain the width (right edge) of the box; leaving a pixel or two for slight “growth” can resolve undesired wrapping.

## Relative Positioning Can Compound Platform Differences

Note the clearly misaligned header in the text box in the following image:



... then all of this text would have moved down as well

**Información turística.** Teléfonos y *webs* de los territorios históricos y comarcas (al comienzo de cada ruta), así como de los municipios. Sorprende la abundancia oceánica en Internet sobre los aspectos turísticos referidos a Euskadi. Se incluyen recomendaciones a la hora de aparcar.

**Museos e iglesias.** Con especial referencia a las visitas guiadas, perfecto complemento de esta guía.

**Dónde comer.** Desde el menú del día más modesto al menú degustación de Arzak. Los menús incluyen bebida, salvo que se especifique otra cosa, e IVA del 8%.

**Dónde dormir.** Hoteles, turismos rurales y agroturismos de caserío, con encanto y personalidad, huyendo si es posible de los ruidos del tráfico.

**Material gráfico.** Los **mapas** son de producción propia, confeccionados al detalle con anotaciones derivadas de los recorridos del autor. Los números que salpican el texto tienen su correspondencia cartográfica. Todas las **fotografías** de esta guía, sin excepción, incluidos museos y obras de arte, fueron realizadas por el autor.

if text would have wrapped to here ...

**BILINGÜISMO FAVORABLE**

Should have used absolute position.

El bilingüismo vigente en Euskadi será recibido con alborozo por el vascovisitante. No hay panel informativo, sin contar la gran mayoría de contestadores automáticos, que no se expresen en euskera y castellano. De entrada, conviene no fiarse de la memoria ante la nomenclatura vascuence (carece de raíz latina; sin acentos ni uves): mejor anotarlo todo. Si escucha decir *¡aupa!,* antes que una alusión al Athletic de Bilbao, es un saludo informal, lo mismo que *¡kaixo!* y *¡epa!* Respecto a la cartelería de los aseos: *Emakumea* (señoras) y *Gizona* (caballeros).

Nº de ruta y color

Localidad o lugar

Información

Identificación

The CSS below was used to position the right-hand column of text ("BILINGUISMO...") at a relative 530px above the end of the left column of text.

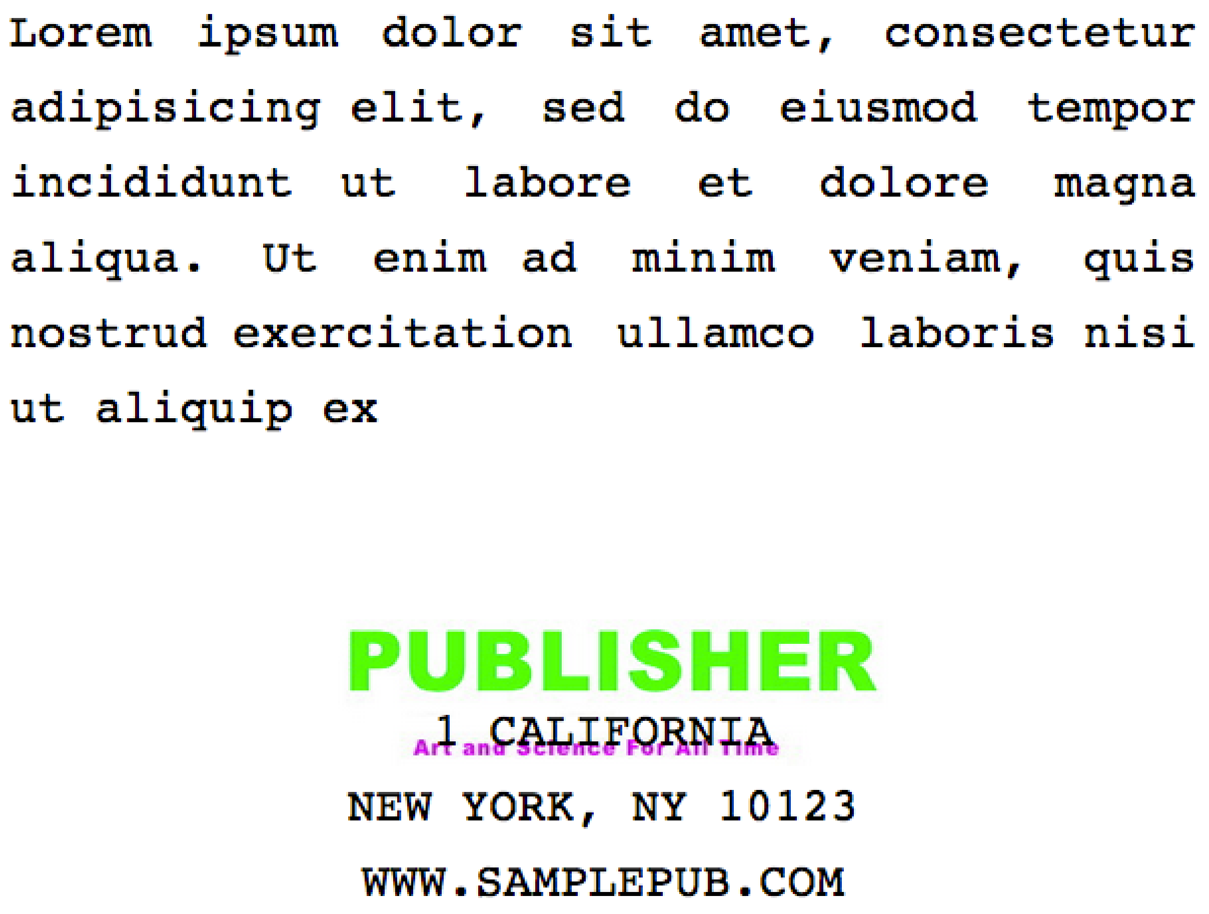
```
.page15a {
        text-align: left;
        position: relative;
        margin-top: -530px;
        margin-left: 661px;
}
```

Apparently, the WebView did a slightly better job of compressing justified text than expected, or slightly more leading per line was compounded. The solution would be to absolutely position the placement of the second column, as follows:

```
.page15a {
        text-align: left;
        position: absolute;
        top: 242px;
        left: 661px;
}
```

## More Relative Positioning

Note the following zoomed image of the bottom of a page from mocked-up content that mirrors actual fixed-layout content.



Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex

PUBLISHER
1 CALIFORNIA
Art and Science For All Time
NEW YORK, NY 10123
WWW.SAMPLEPUB.COM

The text “1 CALIFORNIA” (and all of the shown leading and trailing text, as well as the preceding paragraphs not shown here) is just provided in markup using an embedded font; there is no absolute positioning. By the time the last text on the page is placed, small differences in rendering accumulate and lead to the “1 CALIFORNIA” being placed somewhat too high in relation to the image it overlaps. That is, the content relatively positions each paragraph of text, and then absolutely positions the single image (containing “PUBLISHER” and the smaller “Art and Science For All Time”) into a space left in the text. That level of pixel-to-pixel flow can't be expected to be the same across all browsers (and versions of FreeType). The image should be placed inline with the flow (or the paragraphs should be absolutely positioned). Viewing this content on some browsers will show the undesirable text/image collision, others will not. It is a platform difference that should be accounted before in EPUB Fixed-Layout content.

## EPUB 3 CSS Profile

EPUB 3 has a supported CSS profile (http://idpf.org/epub/30/spec/epub30-contentdocs.html#sec-css-profile), and it is important to use of only CSS properties from within this profile in order for your EPUB content to display correctly. For example, note the following markup:

```
<style type="text/css">
       #contents-img {
       background-image: url("images/1.jpg");
       position: absolute;
       height: 846px;
       width: 600px;
       background-size: 100% auto;
       }
</style>
...
<div id="contents-img"></div>
```

This markup was intended to be used to place all of the images (with, unfortunately, all of the text in the images). The `background-size` property is a proposed addition to CSS 3 Backgrounds and Borders (currently at candidate recommendation level):http://www.w3.org/TR/css3-background/#the-background-size.

However, this property is not part of the EPUB 3 CSS profile. Thus, with the `background-size` property expunged due to not being in the EPUB 3 CSS profile, and as each image was two times larger than the viewport, an EPUB 3-conformant reading systems (correctly) displays only the upper-left portion of the image. The fix would be either to avoid placing the images with the `background-image` property, or to double the size of the viewport to be the same size as the images (so the full image will be used).

# Resources

## IDPF and Other Official Standards

### *EPUB 2*

http://idpf.org/epub/201

### *EPUB 3*

http://idpf.org/epub/30

### *EPUB 3 Fixed-Layout Documents ("FXL")*

http://idpf.org/epub/fxl

## EPUB 3 Sample Documents

http://code.google.com/p/epub-samples/

"This is a repository of EPUB 3.0 sample documents. The collection is intended to showcase features of the EPUB 3 standard—http://idpf.org/epub/30—and to provide testing materials for Reading System developers."

## Books

The IDPF, in collaboration with O'Reilly, has created a comprehensive book on EPUB 3. ***EPUB3 Best Practices: Optimize Your Digital Books*** is now available.

Several chapters are available as free downloads:

***What Is EPUB 3?***—An Introduction to the EPUB Specification for Multimedia Publishing by Matt Garrish
***Accessible EPUB 3***—Best Practices for Creating Universally Usable Content by Matt Garrish

© 2014, the Book Industry Study Group, Inc.

## Blogs

Pigs, Gourds, and Wikis by Liz Castro http://www.pigsgourdsandwikis.com/
Notes & News by Baldur Bjarnason http://www.baldurbjarnason.com/

## Twitter

The #eprdctn hashtag discusses everything about ebook production, and there is often discussion of fixed-layout issues.

# Glossary

***Absolute Positioning*** In CSS, an absolutely positioned object is removed from the flow, and placed using explicit coordinate values (often “top” and “left’). See http://www.w3.org/TR/CSS2/visuren.html#positioning-scheme

***AAL (Advanced Adaptive Layout)*** A draft specification, now called EPUB Adaptive Layout, for an extension to EPUB 3 that would allow complex, magazine-style layouts that would adapt to different screen sizes and orientations. *See* http://idpf.org/epub/pgt/.

***AHL (Advanced/Hybrid Layout)*** A working group of the IDPF will look at ways of extending the capabilities of EPUB 3 by (1) allowing mapping between different renditions of the same content—for example, clicking on a certain portion of a full-page image would bring up a text version of that article—and (2) creating a mechanism for a device to choose between different versions of the same content (the mechanism may be optimized for different screen sizes or device capabilities).

***DRP (Digital Replica Plus)*** Barnes & Noble’s proprietary format for ebooks that consist of an image layer (a magazine page, for example) as well as a text layer, and a mechanism for linking the two versions of the content. *See* http://code.google.com/p/epub-revision/wiki/BNFixedFormat.

***EPIB*** Barnes & Noble’s “subset of the EPUB standard that describes a digital book that can include animated and/or interactive elements.” *See* http://automatastudios.com/barnes-noble-nook-kids-reader/.

***EPUB*** A “distribution and interchange format standard for digital publications and documents. EPUB defines a means of representing, packaging and encoding structured and semantically enhanced Web content…for distribution in a single-file format.” *See* http://idpf.org/epub/30/spec/epub-30-overview.html.

***Folio*** Adobe’s proprietary file format for magazine-like digital publications. Folio (.folio) files are created in InDesign.

***FXL*** Abbreviation for fixed layout used by the IDPF EPUB working group.

***KEPUB*** Kobo’s proprietary version of EPUB. A normal EPUB file is supplemented by a SQLite database that supports some reading system functionality. There may be additional CSS or JS files.

© 2014, the Book Industry Study Group, Inc.

***IDPF (International Digital Publishing Forum)*** The global trade and standards association for electronic publishing, which develops and maintains the EPUB standard.

***JS*** JavaScript, a programming language used to add interactivity to ebooks.

***KF8 (Kindle Format 8)*** Amazon's proprietary file format based on HTML5 and CSS3. It can be used to create fixed-layout ebooks. Note that KF8 is not identical to EPUB 3, although there are some similarities.

***Liquid Layout*** Adobe's term for advanced adaptive layout, as implemented in InDesign CS6.

***Media Overlays*** A subset of SMIL adapted for use in EPUB 3. *See* http://idpf.org/epub/30/spec/epub-30-mediaoverlays.html.

***OFIP*** (Open Format for Interactive Publishing). WoodWing's proprietary format for interactive publications, which was released as a de facto standard, although the page purporting to document this standard redirects to WoodWing's home page.

***SMIL*** As defined by the W3C: "Synchronized Multimedia Integration Language (SMIL, pronounced 'smile') enables simple authoring of interactive audiovisual presentations. SMIL is typically used for 'rich media'/multimedia presentations which integrate streaming audio and video with images, text or any other media type."

© 2014, the Book Industry Study Group, Inc.