XHTML™-Print

Draft 0.56
March 1, 2001

™ XHTML is a trademark of the World Wide Web Consortium.

Table of Contents

1 OVERVIEW

This section is informative.

This document is intended to specify a simple XHTML based data stream suitable for printing as well as display. It is largely based on the W3C's XHTML Basic with the addition of cascading style sheets (CSS). Its targeted usage is for printing in environments with lightweight and other simple clients that do not have the ability to install a printer specific driver. Throughout this document this data stream is called "XHTML-Print."

XHTML-Print is designed to be implementable in low-cost printers that may not have a full-page buffer and that generally print from top-to-bottom, left-to-right with the paper in a portrait orientation. For other printers (i.e. those that print in another direction or orientation) a full-page buffer may be required.

XHTML-Print is not to be used when strict layout consistency and repeatability are required. The design goal of XHTML-Print is to provide a relatively simple, broadly supportable print datastream where content preservation and reproduction are the goal, i.e. "Content is King." More traditional printer datastreams such as PostScript or PCL are more suitable when strict layout control is required.

This document creates a set of conformance criteria for XHTML-Print. It includes style sheet constructs drawn from CSS2 and proposed for CSS3 to provide a strong basis for rich printing results without a detailed understanding of each individual printer's characteristics. It also defines conformance criteria for an optional extension set targeted at photo printing, the XHTML-Print Photo-Imaging Extension.

2 REFERENCES

This section is informative.

The following definitions and references are used throughout the document.

1. XHTML™ 1.0: The Extensible HyperText Markup Language. A reformulation of HTML 4.0 as an XML application. See http://www.w3.org/TR/xhtml1
2. XHTML™ Basic: A subset of XHTML 1.0 that includes a reduced set of functions. See http://www.w3.org/TR/xhtml-basic
3. Modularization of XHTML™: A document which an abstract modularization of XHTML and an implementation of the abstraction using XML Document Type Definitions (DTDs). This modularization provides a means for subsetting and extending XHTML, a feature needed for extending XHTML's reach onto emerging platforms. See http://www.w3.org/TR/xhtml-modularization
4. XML 1.0: The Extensible Markup Language (XML) is a subset of SGML that is to be served, received, and processed on the Web in the way that is not possible with HTML. XML has been designed for ease of implementation and for interoperability with both SGML and HTML. See http://www.w3.org/TR/REC-xml
5. CSS 1.0: Cascading Style Sheets version 1.0 is a simple style sheet mechanism that allows authors and readers to attach style (e.g. fonts, colors and spacing) to HTML documents. See http://www.w3.org/TR/REC-CSS1
6. CSS 2.0: Cascading Style Sheets version 2.0 build on CSS1 and, with very few exceptions, all valid CSS1 style sheets are valid CSS2 style sheets. CSS2 supports media-specific style sheets so that authors may tailor the presentation of their documents to visual browsers, aural devices, printers, braille devices, handheld devices, etc. CSS2 also adds content positioning, downloadable fonts, table layout, features for internationalization, automatic counters and numbering, and some properties related to user interface. See http://www.w3.org/TR/REC-CSS2/
7. "Namespaces in XML", T. Bray, D. Hollander, A. Layman, 14 January 1999. XML namespaces provide a simple method for qualifying names used in XML documents by associating them with namespaces identified by URI. Available at: http://www.w3.org/TR/REC-xml-names.
8. "Simple Object Access Protocol (SOAP) 1.1" SOAP is a lightweight XML-based protocol for exchange of information in a decentralized, distributed environment. It is a submission to the W3C and is available at http://www.w3.org/TR/SOAP.
9. "Introduction to CSS3" is an introduction and roadmap of the work in progress on CSS level 3. At this time, it is available to W3C members only from: http://www.w3.org/Style/Group/css3-src/css3-roadmap/
10. "RFC 2045 - Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", N. Freed, N. Borenstein, Section 6.8. "Base64 Content-Transfer-Encoding" is of particular interest It is available at http://info.internet.isi.edu/in-notes/rfc/files/rfc2045.txt
11. "XML Schema Part 2: Datatypes W3C Working Draft 22 September 2000", P. Biron, A Malhotra, available at http://www.w3.org/TR/xmlschema-2/

12. “JPEG File Interchange Format, version 1.02, September 1, 1992”, C-Cube Microsystems. Available from ftp://ftp.uu.net/graphics/jpeg/jfif.ps.gz

3 XHTML-PRINT TAGS

This section is normative.

3.1 Tags Required by XHTML-Print

The World Wide Web Consortium has defined a subset of XHTML 1.0 that is targeted to small format devices such as PDAs and cellular telephones. The definition of XHTML-Basic is therefore useful to be examined as a starting point for the definition of XHTML-Print. XHTML-Print is a proper superset of XHTML Basic in the set of tags required.

The Modularization of XHTML [Reference 3 on page 5] is a decomposition of XHTML 1.0 and by reference HTML 4.0 into a collection of abstract modules that provide specific types of functionality. XHTML-Print is defined, in part, by inclusion of a set of these modules. As such, the non-CSS portion of XHTML-Print includes the following modules:

1. Core Modules
   - Structure Module
     body, head, html, title
   - Text Module
     abbr, acronym, address, blockquote, br, cite, code, dfn, div, em, h1, h2, h3, h4, h5, h6, kbd, p, pre, q, samp, span, strong, var
   - Hypertext Module
     a
   - List Module
     dl, dt, dd, ol, ul, li
2. Text Extension Modules
   - Presentation
     b, big, hr, i, small, sub, sup, tt
3. Table Modules
   - Basic Table Module
     caption, table, td, th, tr
4. Image Module
   img
5. Forms Modules
   - Basic Forms Module
     form, input, label, select, option, textarea
6. Metainformation Module
   meta
7. Style Sheet Module
   style
8. Style Attribute Module
9. Link Module
   link
10. Base Module
    base

11. Object Module

object

param

3.2 Restrictions on XHTML by XHTML-Print

XHTML-Print also restricts some usage of common XTHML 1.0 tags in the same way that XHTML Basic does:

- Nesting of Tables is NOT supported
- Frames are NOT supported

3.3 CSS Conformance

See section 5.3.2 for CSS conformance requirements for XHTML-Print conforming implementations.

4 APPLICATION OF XHTML/CSS FOR XHTML-PRINT

This section is normative.

XHTML-Print inherits all the structure, encoding and other basic infrastructure specified by XHTML. The following sections describe and clarify the application and usage restrictions of XHTML-Print in this environment.

4.1 Recommended attributes on the <img> and <object> tags

Because many printers create the page in a serial manner from top to bottom, it is important for the printer to know the size of images before retrieving the image data itself. This information is then used to create portions of the page layout.

Therefore, the sender is strongly encouraged to include the height and width attributes either within the <*img*> or the <*object*> tag, or within an associated style sheet rule. These attributes may be expressed as percentages within the <*img*> or the <*object*> tag, or may use the standard absolute or relative units within the CSS rule. Percentages are relative to the parent element and not the page width or printable area. Pixel units should be avoided, because the resultant size may vary markedly, depending on the native resolution of the printer or displaying device.

This document specifies only one mandatory image format, baseline JPEG as defined in 12 on page 4. See Appendix A for a description of JPEG decoder requirements. Printers are not required to support:

- Embedded Thumbnails
- Rotation
- Progressive rendering

within the JFIF files.

4.2 Style Sheets

Conforming XHTML-Print printers shall support both in-line and referenced style sheets within the <*style*> tag or <*link*> tag in the <*head*> of a document. Conforming XHTML-Print printers shall also support the style attributes (i.e. in-line style) when used within the various elements of the documents. Normal cascading rules apply. See section 4.8 of [1} referenced on page 4 for special cases when the style section includes special characters such as “&” and “<”.

4.3 Page Breaks

Because of the differences in displaying in a browser versus a paged media device like a printer, the data source may want to have control of the locations of page breaks. Therefore, a single means is needed to cause a *page-break* to occur.

Example Page Break using CSS2

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/TR/xhtml1">
 <head>
  <title> Testing Page Breaks </title>
<style>
.pagebreak { page-break-after: always }
</style>
</head>
 <body>
<p>Page 1 Text</p>
<br class="pagebreak" />
<p>Page 2 Text</p>
 </body>
</html>
```

As such, Conforming implementations shall support the CSS2 ***page-break*** properties. Usage of this property is allowed with <br> as well as all other appropriate tags, e.g. <p>, <div>, <h1>, <ul>, etc.

If ***page-break-inside avoid*** is specified for a long element and the printer is unable to buffer the entire element before committing it to paper, it should force a page break to occur before the long element and begin the element starting at the top of the next page. If the long element starts at the top of a page and exceeds the page length, the printer shall print as much as possible on the first page and then resume that element on the next and subsequent pages as required to preserve the content. A printer is not required to perform scaling to fit the long element on a single page.

4.4 Page Size and Orientation

(The following is a summary and partial extraction from the CSS2 document)

ISSUE: What is the relationship between Page Size and Printable Area?
Page size and orientation are controlled using the @page rules from CSS2. Specifically, the size property is applied to @page to set both size and orientation. The margin properties ('margin-top', 'margin-bottom', 'margin-left', 'margin-right', and 'margin') as defined by CSS2 also apply within the page context. Page size and orientation provided in the XHTML-Print datastream will override similar attributes contained within any commands and/or attributes provided when creating the print job itself.

4.4.1 Size Property

Usage:

```
@page {
    size: auto;  /* auto is the initial value and uses the default paper */
        }
```

'size'

Value: <length>{1,2} | auto | portrait | landscape | inherit
Initial: auto
Applies to: the page context
Inherited: N/A
Percentages: N/A
Media: paged

This property specifies the size and orientation of a page box. The size of a page box may either be "absolute" (fixed size) or "relative" (scalable, i.e., fitting available sheet sizes). Relative page boxes allow printers to scale a document and make optimal use of the target size. Three values for the 'size' property create a relative page box:

- auto: The page box will be set to the size and orientation of the target sheet.
- landscape: Overrides the target's orientation. The page box is the same size as the target, and the longer sides are horizontal.
- portrait: Overrides the target's orientation. The page box is the same size as the target, and the shorter sides are horizontal.

4.4.2 Margin Property

The margin property is supported as specified in CSS2, clause 8.3.

4.4.3 Examples

In the following example, the outer edges of the page box will align with the target. The percentage value on the 'margin' property is relative to the target size so if the target sheet dimensions are 21.0cm x 29.7cm (i.e., A4), the margins are 2.10cm and 2.97cm.

```
@page {
  size: auto;   /* auto is the initial value */
  margin: 10%;
        }
```

Length values for the 'size' property create an absolute page box. If only one length value is specified, it sets both the width and height of the page box (i.e., the box is a square). Since the page box is the initial containing block, percentage values are not allowed for the 'size' property.

For example:

```
@page {
  size: 8.5in 11in portrait;      /* width height */
          }
```

The above example set the width of the page box to be 8.5in and the height to be 11in. The page box in this example requires a target sheet size of 8.5"x11" or larger. Printers may allow users to control the transfer of the page box to the sheet (e.g., rotating an absolute page box that's being printed).

4.4.4 Rendering page boxes that do not fit a target sheet

If a page box does not fit the target sheet dimensions, the printer may choose (in order of preference) to:

- Rotate the page box 90° if this will make the page box fit.
- Scale the page to fit the target.
- Reformat the page (including "spilling" onto another sheet)
- Clip (least preferred)

The printer may consult the user before performing these operations. Lacking "access" to the user, it may simply make a decision on its own.

4.4.5 Positioning the page box on the sheet

When the page box is smaller than the target size, the user agent is free to place the page box anywhere on the sheet. However, it is recommended that the page box be centered on the sheet since this will align double-sided pages and avoid accidental loss of information that is printed near the edge of the sheet.

4.5 Running headers and Footers

A means is needed to create a *running-header* and a *running-footer* on the printed page. Current work in progress by the W3C on paged media defines a very robust method for adding margin boxes to the top, bottom, left and right of the page. A reduced set from the CSS3 proposal is employed by XHTML-Print, using top and bottom margin boxes via the @page rules method.

ISSUE: ***CSS3 progress should be tracked and mirrored for this feature.***

Utilizing the terminology of CSS2 and CSS3, a 'margin box' is defined in conjunction with the 'page box' and 'page area' (as shown in Figure 1) to create an area into which *running-header* and *running-footer* text can be inserted.

CSS3 proposes the ability to left-align, right-align and center the text horizontally as well as methods to top-align, bottom-align and center the text vertically within the margin boxes. For XHTML-Print conforming implementations vertical controls are not required to be supported. Instead, for XHTML-Print conforming implementations, the *running-header* text may be top aligned in the margin box and the *running-footer* text may be bottom aligned in the margin box.

CSS3 proposes methods for the printing device to automatically include:

- page number
- total pages in the document
- date
- time
- file name

into the *running-header* and *running-footer.* XHTML-Print conforming implementations are only required to support inserting a page number using the "counter(pages)" object. If required, the sending appliance must provide the other information within the text string to be printed in the margin box.

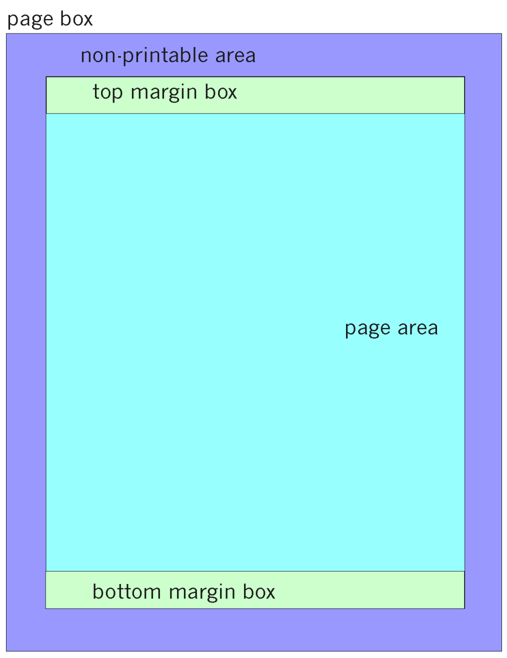

Figure 1

The following are sample XHTML/CSS fragments used to create *running-headers* and *running-footers.*

```
<style>
@page {
       @top{font-family: Helvetica, Arial, sans-serif
            font-size: 150%
            font-weight: bolder
            text-align: left
            content: "XHTML-Print: A Proposal --- August 25, 2000";
           }
       }
</style>
```

The above example creates a running header that is left aligned at 150% of normal font size and bold in Helvetica, Arial or the default san-serif font whichever is available.

```
<style>
@page {
       @bottom{font-family: Times, Palatino, serif
               font-size: 80%
               font-weight: normal
               text-align: center
               content: "Page " counter(pages);
              }
       }
</style>
```

The above example creates a running footer such as "Page 14" centered on the page in a font 80% of normal size in Times, Palatino or the default serif font whichever is available.

4.6 Image Data

In traditional web-based applications of XHTML, image data is contained in a separate file on a web server that the user agent retrieves.

However, there are circumstances where it is desirable to include the image data along with the rest of the print data. Some low cost, resource constrained clients may want to include images in their print output but cannot afford to include a server. Some print applications may require that all the print data can be encapsulated in a single file for transportability, avoiding firewall issues, etc.

See Appendix B for discussion of a method allowing both XHTML-Print and associated image data to be collected into a single file or data stream.

4.7 Side-by-Side Images

Low-cost printers today often have very little memory into which page data can be stored before being printed. As such, they must build and print the page in swaths on the fly from the top of the page to the bottom. To enable the use of XHTML-Print in these low cost printers, some restrictions on the order of images contained in the XHTML-Print data stream must be added.

1. If two or more images will be even partially side-by-side on the printed page they should be included by reference (<img src="http://10.10.10.2/images/logo.jpg"> or <object data=http://10.10.10.2/images/logo.jpg>) rather than included in-line. (See Appendix B). This allows the printer to get chunks of the image, as it needs it, as it prints down the page.

2. An XHTML-Print conforming printer lacking sufficient buffer space to hold multiple side-by-side images may choose to reformat the layout of the page to preserve content. Printers shall attempt to preserve content when encountering side-by-side images that may be impossible to print as specified within the XHTML-Print. Discarding the second and subsequent of the side-by-side images should be avoided unless preservation of content is best achieved by doing so. Other than attempting to best preserve content, this specification DOES NOT mandate any specific behavior when encountering this situation. Clients providing the images in-line should order them from left-to-right, top-to-bottom unless the print direction is known to be otherwise.

5 CONFORMANCE

This section is normative.

5.1 XHTML Document Type Conformance

A Conforming XHTML-Print document is a document that requires only the facilities described as mandatory in this specification. Such a document must meet all of the following criteria:

1. The document must validate against the DTD found in C and conform to the constraints expressed in section 3.2.
2. The root element of the document must be <html>.
3. The name of the default namespace on the root element must be the XHTML namespace name, http://www.w3.org/1999/xhtml.
4. There must be a DOCTYPE declaration in the document prior to the root element. If present, the public identifier included in the DOCTYPE declaration must reference the DTD found in C   using its Formal Public Identifier. The system identifier may be modified appropriately.

```
<!DOCTYPE html PUBLIC "-//PWG//DTD XHTML-Print 1.0//EN"
    "http://www.pwg.org/xhtml-print/xhtml-print10.dtd">
```

5.2 Client Conformance

1. Clients shall produce a well-formed XHTML document as defined in 1 on page 4.
2. Beyond number 1 above, clients are not required to use more of the XHTML-Print tags or Style Sheet attributes than necessary to get the desired output.

5.3 Printer Conformance

5.3.1 Formatting/Rendering Rules

1. In order to be consistent with the XML 1.0 Recommendation [see page 5], the printer must parse and evaluate an XHTML document to determine if the document is well formed. If the printer claims to be a validating printer, it must also validate documents against their referenced DTDs according to XML. Validation is not required to claim conformance to this standard. A printer may "flush" or otherwise reject a non-conforming XHTML-Print document.
2. When the printer claims to support facilities defined within this specification or required by this specification through normative reference, it must do so in ways consistent with the facilities' definition.
3. When a printer processes an XHTML document as generic XML, it shall only recognize attributes of type ID (e.g. the id attribute on most XHTML elements) as fragment identifiers.
4. Images:
    - If a printer encounters an image in a format it does not support, it will reserve the space specified by the height and width attributes optionally by drawing a box around this space of the size specified for the image.

    - If the image format is not supported and the height and width attributes were omitted, the image is omitted and no space is reserved.
    - If the image format is supported and the height and width attributes were omitted, the printer may choose to omit the image from the page.
5. If a printer encounters an element it does not recognize, it should render the element's content as if the element and its end tag were not present at all. Printers may chose not to render content within elements defined by XHTML, HTML or deprecated from HTML which is obviously not intended to be rendered, e.g. <script>.
6. If a printer encounters an attribute it does not recognize, it must ignore the entire attribute specification (i.e., the attribute and its value).
7. If a printer encounters an attribute value it doesn't recognize, it must use the default attribute value.
8. If a printer encounters an entity reference, e.g. "&#X007;", (other than one of the predefined entities) for which the Printer has processed no declaration (which could happen if the declaration is in the external subset which the Printer hasn't read), the entity reference should be rendered as the characters (starting with the ampersand and ending with the semi-colon) that make up the entity reference.
9. When rendering content, printers that encounter characters or character entity references that are recognized but not renderable should display the document in such a way that it is obvious to the user that normal rendering has not taken place.
10. The following characters are defined in XML as whitespace characters: Space (&#x0020;) Tab (&#x0009;) Carriage return (&#x000D;) Line feed (&#x000A;). The XML processor normalizes different system's line end codes into one single line-feed character that is passed up to the application. The XHTML printer in addition, must treat the following characters as whitespace: Form feed (&#x000C;) Zero-width space (&#x200B;) In elements where the 'xml:space' attribute is set to 'preserve', the printer must leave all whitespace characters intact (with the exception of leading and trailing whitespace characters, which should be removed). Otherwise, whitespace is handled according to the following rules:
    - All whitespace surrounding block elements should be removed.
    - Comments are removed entirely and do not affect whitespace handling. One whitespace character on either side of a comment is treated as two white space characters.
    - Leading and trailing whitespace inside a block element must be removed.
    - Line feed characters within a block element must be converted into a space (except when the 'xml:space' attribute is set to 'preserve').
    - A sequence of white space characters must be reduced to a single space character (except when the 'xml:space' attribute is set to 'preserve').
11. With regard to rendition, the printer should render the content in a manner appropriate to the language in which the content is written. Additionally,
    - In languages whose primary script is Latinate, the ASCII space character is typically used to encode both grammatical word boundaries and typographic whitespace.
    - In languages whose script is related to Nagari (e.g., Sanskrit, Thai, etc.), grammatical boundaries may be encoded using the ZW 'space' character, but will not typically be represented by typographic whitespace in rendered output.
    - Languages using Arabiform scripts may encode typographic whitespace using a space character, but may also use the ZW space character to delimit 'internal' grammatical

    boundaries (what look like words in Arabic to an English eye frequently encode several words, e.g. 'kitAbuhum' = 'kitAbu-hum' = 'book them' == their book).
  - Languages in the Chinese script tradition typically neither encode such delimiters nor use typographic whitespace in this way. Whitespace in attribute values is processed according to [XML].

5.3.2 Printer Requirements

- A conforming printer shall support all XHTML Modules listed in clause 3.1 on page 7.
- A conforming printer shall print a static version of a form using default values as specified in the form.
- A conforming printer shall support the following CSS constructs:

1. Block item properties:
   - Font (font-family, font-style, font-variant, font-weight, font-size)
   - Color (RGB values and the 16 named colors defined by section 4.3 in Modularization of XHTML (Reference 3 on page 4); support for system colors is not required.)
   - Text decoration (underline, overline, linethrough)
   - Text align (left, right, center)
   - Text indent
   - Line Height
2. Classification Properties
   - White-space
   - List-style-type (none, disc, circle, square, decimal, lower-alpha, upper-alpha)
   - List-style-position
3. Units
   - em
   - ex
   - mm
   - inches
   - points
   - percent
4. @media print
5. @page rules
   - size
   - margin
   - :left
   - :right
   - :first
   - crop is NOT required to be supported
   - named pages for content placement control is NOT required to be supported
6. Page Break Properties as applied to <br> and block elements (e.g. <p></p>, <ol></ol>, etc.)
   - Page-break-before
   - Page-break-after
   - Page-break-inside

The page-header and page-footer constructs from the CSS3 proposal shall be supported as described in Section 4.5.

5.4 Advanced Layout Extension

To further support print applications requiring more exacting page layout (*e.g.,* photo album pages), the following additional style sheet properties and image format are required to be supported in an optional, discoverable Advanced Layout Extensions device. If support for this extension is indicated, all of the following must be supported:

1. Box Properties
    - margin-top, margin-bottom, margin-right, margin-left, margin
    - padding-top, padding-bottom, padding-right, padding-left, padding
    - border-top-width, border-bottom-width, border-right-width, border-left-width, border-width
    - border-top-color, border-bottom-color, border-right-color, border-left-color, border-color
    - border-top, border-bottom, border-right, border-left, border
    - position (static, relative, absolute, fixed)
    - box offsets (top, bottom, left, right)
    - clip

2. Image File Format
    - See Appendix A for a description of additional JPEG decoding requirements for this extension.

The following is an example using absolute positioning with image data:

```
<style>
.picture1 {
                position: absolute;
                top: 25mm;
                left: 25mm;
                padding-top: 10mm;
                width: 30mm;
                height: 30mm;
                clip: rect(10mm, 30mm, 30mm, 0mm)
            }
</style>
…
<div class=”picture1”>
<img src=”some_file.jpg” />
</div>
```

6 AUTHORS

Authors' Addresses:

Don Wright
Lexmark International
don@lexmark.com
+1 859-232-4808

Melinda Grant
Hewlett-Packard
melinda_grant@hp.com
+1 360-212-3544

Peter Zehler
Xerox
peter.zehler@usa.xerox.com
+1 716-265-8755

Jun Fujisawa
Canon
fujisawa.jun@canon.co.jp
+81 44-733-6111

A JPEG Decoder Requirements

A.1 Introduction

A.1.1 Intent

The intent of this appendix is to describe recommended behaviors for JPEG decoders in XHTML-print devices. Behaviors for both minimal printers and photo printers are described. Many of the behaviors described in this document follow directly from language already present in the relevant JPEG standards, but are repeated here to emphasize their importance.

A.1.2 Objectives

The decoder behaviors described in this document are intended to minimize implementation complexity, while retaining maximum compatibility with existing JPEG files. In particular, these recommendations seek to ensure compatibility with both EXIF and baseline JFIF (i.e. the subset of JFIF files that use only baseline JPEG processes). Support for JPEG streams using non-baseline processes, such as arithmetic coding or progressive coding, is not mandated for XHTML-Print compliance.

A.2 Behaviors of Minimal Printers

This section describes behaviors of JPEG decoders for minimal XHTML-Print implementations.

A.2.1 JPEG processes

A JPEG decoder for an XHTML-Print server must support all baseline JPEG processes as defined in section 3.11 of ISO DIS 10918-1 / CCITT Rec T.81, except for 2- and 4-component images. These processes include greyscale and 3-component images, 8-bit/component sample depth, Huffman entropy coding, 444, 422, 411, and 400 subsampling modes, and sequential (i.e. non-progressive) scan."

A.2.2 Handling of APPx markers

Baseline decoders may ignore application-specific markers, such as the JFIF APP0 marker and the EXIF APP1/APP2 markers. This will cause all images to print in an un-rotated orientation, with image size as specified in the JPEG SOF marker. A JPEG decoder for a minimal printer must not fail as a consequence of encountering an unsupported APPx marker (i.e. all such markers must be correctly parsed, even if they are ignored).

A.2.3 Color Management

This section describes a recommended color management approach for minimal JPEG printers

A.2.3.1 Greyscale Images

Sample values in a grayscale (single-component) JPEG image shall be converted to the sRGB color space by setting

$$R_{out} = G_{out} = B_{out} = Gray_{in}$$

A.2.3.2 Color Images

Sample values in 3-component JPEG images shall be interpreted as YCbCr samples, as would be obtained by applying the matrices described in ITU BT.601 to sRGB input data.

A.3 XHTML-Print Advanced Layout Extension

This section describes behaviors of JPEG decoders for XHTML-Print devices which support the XHTML-Print Advanced Layout Extension, an optional feature block. The behaviors described below should be interpreted as "in addition to" those described in section 2 (the requirements for minimal XHTML-Print devices).

A.3.1 Handling of EXIF APP1 and APP2 Markers

A JPEG decoder for an XHTML-Print implementation which supports the Advanced Layout Extension must decode the TIFF IFDs embedded in the EXIF APP1 and APP2 markers, as described in Section 2.6.4 of JEIDA-49-1998. The following IFDs must be fully supported (i.e. they may not be ignored).

Tag Name	Field Name	Description
Orientation of Image	Orientation	Sets image orientation in 90-degree increments, and enables transposition.

B INLINE IMAGE DATA

B.1 Introduction

B.1.1 Intent

The intent of this appendix is to describe recommended behaviors for including XHTML-Print and associated image data in a single data stream or file. (See Section 4.6.)

B.1.2 Objectives

- ♦ Minimize image data size
- ♦ No or minimal additional encoding / decoding of image data required
- ♦ Enable juxtaposition between image data and associated XHTML-Print content
  - Many printers are unable to buffer significant amounts of page content data, so the image data must be printed more or less as it is received. This implies the image data must be sent near the related XHTML-Print content, so that layout and printing can occur without extensive data buffering.
- ♦ Minimize complexity
- ♦ Leverage existing standard capabilities

B.2 Multipart MIME Related Method

This method uses multipart MIME types to split the XHTML-print data into related parts with the JPEG binary files interspersed close to where the image is to be printed. Instead of using "Content-type: Multipart/mixed;" this method uses "Content-type: Multipart/related" which indicates that all of the parts are intended to stay together as a single document (refer to http://www.ietf.org/rfc/rfc2387.txt for more details).

The multiple parts of the document are split loosely along boundaries where JPEG data needs to be included. The Multipart/related option is used on Content-type for the whole package to indicate that all of the parts are related in a single document.

Example: `Content-type:  Multipart/related; boundary: A12$GT`

The intent of this parameter is to indicate that the XHTML-print document is fragmented and the pieces must be put together to make a whole document. Each of the fragmented parts includes a Content-type to indicate whether it is XHTML-print or binary image data:

```
Content-type:  text/xhtml-print+xml/partial
Content-type:  image/jpeg
```

ISSUE: The Working Group still needs to get "/partial" approved as a mime type.

The following example shows how the different parts of an XHTML-print document could be fragmented to include a jpeg image.

```
POST / HTTP 1.1
Content-type: multipart/related; boundary: A12$GTq7299v6

A12$GTq7299v6
Content-type: text/xhtml-print+xml/partial ;charset="utf-8"
Content-length: nnn

<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/TR/xhtml1">
 <head>
  <title> Testing Page Breaks
</title>
<style>
.pagebreak { page-break-after: always }
</style>
</head>
 <body>
<p>Page 1 Text</p>
<br class="pagebreak" />
<p>Page 2 Text</p>

<object declare=”declare”
 height=”20 mm” width=”20 mm”
 type=”image/jpeg”
 id=”image_1.jpeg”
</object>

<p>
This example shows that the JPEG image is included as part of the
multipart mimetype with multiple xhtml-print documents that are split
along page boundaries.
</p>

A12$GTq7299v6    {this is the boundary indicator}
Content-Location: image_1.jpeg
Conten-id: <97116092511xyz@foo.bar.net>
Content-type: image/jpeg
Content-length:65538

...65538 bytes of jpeg binary ...

A12$GTq7299v6
Content-type: text/xhtml-print+xml/partial ;charset="utf-8"
Content-length: nnn

<p>
This example shows that the xhtml-print document can be split into
multiple parts.
</p>
</body>
</html>
```

C XHTML-Print DTD and Modules

This section contains the pieces of the XHTML-Print DTD that are unique to XHTML-Print. The remaining entities and modules are as specified in reference 3 on page 5.

The following should be used from Modularization of XHTML:

1. xhtml-attribs-1.mod
2. xhtml-base-1.mod
3. xhtml-basic-form-1.mod
4. xhtml-basic-table-1.mod
5. xhtml-blkphras-1.mod
6. xhtml-blkpres-1.mod
7. xhtml-blkstruct-1.mod
8. xhtml-charent-1.mod
9. xhtml-datatypes-1.mod
10. xhtml-framework-1.mod
11. xhtml-hypertext-1.mod
12. xhtml-image-1.mod
13. xhtml-inlphras-1.mod
14. xhtml-inlpres-1.mod
15. xhtml-inlstruct-1.mod
16. xhtml-inlstyle-1.mod
17. xhtml-lat1.ent
18. xhtml-link-1.mod
19. xhtml-list-1.mod
20. xhtml-meta-1.mod
21. xhtml-notations-1.mod
22. xhtml-object-1.mod
23. xhtml-param-1.mod
24. xhtml-pres-1.mod
25. xhtml-qname-1.mod
26. xhtml-special.ent
27. xhtml-struct-1.mod
28. xhtml-style-1.mod
29. xhtml-symbol.ent
30. xhtml-text-1.mod

C.1 XHTML-Print 1.0 DTD

```
<!-- ....................................................................... -->
<!-- XHTML-Print 1.0 DTD  .................................................. -->
<!-- file: xhtml-print10.dtd
-->

<!-- XHTML-Print 1.0 DTD
```

```
     This is XHTML-Print 1.0, a variant of XHTML Basic for printing.

     Copyright 2001 Printer Working Group. All Rights Reserved.

     Author: Jun Fujisawa <fujisawa.jun@canon.co.jp>
     Revision: Version 1.1, February 26, 2001.

-->
<!-- This is the driver file for version 1.0 of the XHTML-Print DTD.

     Please use this formal public identifier to identify it:

         "-//PWG//DTD XHTML-Print 1.0//EN"
-->
<!ENTITY % XHTML.version  "-//PWG//DTD XHTML-Print 1.0//EN" >

<!-- Use this URI to identify the default namespace:

         "http://www.w3.org/1999/xhtml"
-->
<!ENTITY % NS.prefixed "IGNORE" >
<!ENTITY % XHTML.prefix "" >

<!-- Reserved for use with the XLink namespace:
-->
<!ENTITY % XLINK.xmlns "" >
<!ENTITY % XLINK.xmlns.attrib "" >

<!-- reserved for future use with document profiles -->
<!ENTITY % XHTML.profile "" >

<!-- Bidirectional Text features
     This feature-test entity is used to declare elements
     and attributes used for bidirectional text support.
-->
<!ENTITY % XHTML.bidi "IGNORE" >

<!-- :::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->

<!ENTITY % xhtml-events.module "IGNORE" >
<!ENTITY % xhtml-bdo.module "%XHTML.bidi;" >

<!-- Document Model Module  .................................... -->
<!ENTITY % xhtml-model.mod
     PUBLIC "-//PWG//ENTITIES XHTML-Print 1.0 Document Model 1.0//EN"
            "xhtml-print10-model-1.mod" >

<!-- Modular Framework Module  .................................. -->
<!ENTITY % xhtml-framework.mod
     PUBLIC "-//W3C//ENTITIES XHTML Modular Framework 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-framework-1.mod" >
%xhtml-framework.mod;

<!-- Text Module (required)  ................................... -->
<!ENTITY % xhtml-text.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Text 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-text-1.mod" >
%xhtml-text.mod;

<!-- Hypertext Module (required) ................................ -->
<!ENTITY % xhtml-hypertext.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Hypertext 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-hypertext-1.mod" >
%xhtml-hypertext.mod;

<!-- Lists Module (required)  .................................. -->
<!ENTITY % xhtml-list.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Lists 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-list-1.mod" >
%xhtml-list.mod;
```

```
<!-- Document Structure Module (required)  ...................... -->
<!ENTITY % xhtml-struct.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Document Structure 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-struct-1.mod" >
%xhtml-struct.mod;

<!-- :::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->

<!-- Presentation Module  ....................................... -->
<!ENTITY % xhtml-pres.module "INCLUDE" >
<![%xhtml-pres.module;[
<!ENTITY % xhtml-pres.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Presentation 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-pres-1.mod" >
%xhtml-pres.mod;]]>

<!-- Image Module  ............................................. -->
<!ENTITY % xhtml-image.module "INCLUDE" >
<![%xhtml-image.module;[
<!ENTITY % xhtml-image.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Images 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-image-1.mod" >
%xhtml-image.mod;]]>

<!-- Tables Module ............................................. -->
<!ENTITY % xhtml-table.module "INCLUDE" >
<![%xhtml-table.module;[
<!ENTITY % xhtml-table.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Basic Tables 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-basic-table-1.mod" >
%xhtml-table.mod;]]>

<!-- Forms Module  ............................................. -->
<!ENTITY % xhtml-form.module "INCLUDE" >
<![%xhtml-form.module;[
<!ENTITY % xhtml-form.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Basic Forms 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-basic-form-1.mod" >
%xhtml-form.mod;]]>

<!-- Style Sheet Module  ........................................ -->
<!ENTITY % xhtml-style.module "INCLUDE" >
<![%xhtml-style.module;[
<!ENTITY % xhtml-style.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Stylesheets 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-style-1.mod" >
%xhtml-style.mod;]]>

<!-- Style Attribute Module  ................................... -->
<!ENTITY % xhtml-inlstyle.module "INCLUDE" >
<![%xhtml-inlstyle.module;[
<!ENTITY % xhtml-inlstyle.mod
     PUBLIC "-//W3C//ENTITIES XHTML Inline Style 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-inlstyle-1.mod" >
%xhtml-inlstyle.mod;]]>

<!-- Link Module  ............................................... -->
<!ENTITY % xhtml-link.module "INCLUDE" >
<![%xhtml-link.module;[
<!ENTITY % xhtml-link.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Link Element 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-link-1.mod" >
%xhtml-link.mod;]]>

<!-- Metainformation Module  ................................... -->
<!ENTITY % xhtml-meta.module "INCLUDE" >
<![%xhtml-meta.module;[
<!ENTITY % xhtml-meta.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Metainformation 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-meta-1.mod" >
%xhtml-meta.mod;]]>
```

```
<!-- Base Module  ............................................. -->
<!ENTITY % xhtml-base.module "INCLUDE" >
<![%xhtml-base.module;[
<!ENTITY % xhtml-base.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Base Element 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-base-1.mod" >
%xhtml-base.mod;]]>

<!ENTITY % xhtml-param.module "INCLUDE" >
<![%xhtml-param.module;[
<!ENTITY % xhtml-param.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Param Element 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-param-1.mod" >
%xhtml-param.mod;]]>

<!-- Object Module  ........................................... -->
<!ENTITY % xhtml-object.module "INCLUDE" >
<![%xhtml-object.module;[
<!ENTITY % xhtml-object.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Embedded Object 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-object-1.mod" >
%xhtml-object.mod;]]>

<!-- end of XHTML-Print 1.0 DTD  ........................................ -->
<!-- ................................................................... -->
```

C.2 XHTML-Print 1.0 Document Model Module

```
<!-- ................................................................... -->
<!-- XHTML-Print 1.0 Document Model Module  ............................ -->
<!-- file: xhtml-print10-model-1.mod
-->

<!-- XHTML-Print 1.0 Document Model Module

     This is XHTML-Print 1.0, a variant of XHTML Basic for printing.

     Copyright 2001 Printer Working Group. All Rights Reserved.

     Author: Jun Fujisawa <fujisawa.jun@canon.co.jp>
     Revision: Version 1.1, February 26, 2001.

-->
<!-- This is the content model file for version 1.0 of the XHTML-Print DTD.

     Please use this formal public identifier to identify it:

         "-//PWG//ENTITIES XHTML-Print 1.0 Document Model 1.0//EN"
-->

<!-- :::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->

<!-- Optional Elements in head  ............................... -->

<!ENTITY % HeadOpts.mix
     "( %meta.qname; | %link.qname; | %object.qname; | %style.qname; )*" >

<!-- Miscellaneous Elements  .................................. -->

<!ENTITY % Misc.class "" >

<!-- Inline Elements  ......................................... -->

<!ENTITY % InlStruct.class "%br.qname; | %span.qname;" >

<!ENTITY % InlPhras.class
     "| %em.qname; | %strong.qname; | %dfn.qname; | %code.qname;
```

```
      | %samp.qname; | %kbd.qname; | %var.qname; | %cite.qname;
      | %abbr.qname; | %acronym.qname; | %q.qname;" >

<!ENTITY % InlPres.class
     "| %tt.qname; | %i.qname; | %b.qname; | %big.qname;
      | %small.qname; | %sub.qname; | %sup.qname; " >

<!ENTITY % I18n.class "" >

<!ENTITY % Anchor.class "| %a.qname;" >

<!ENTITY % InlSpecial.class "| %img.qname; | %object.qname;" >

<!ENTITY % InlForm.class
     "| %input.qname; | %select.qname; | %textarea.qname;
      | %label.qname;"
>

<!ENTITY % Inline.extra "" >

<!ENTITY % Inline.class
     "%InlStruct.class;
      %InlPhras.class;
      %InlPres.class;
      %Anchor.class;
      %InlSpecial.class;
      %InlForm.class;
      %Inline.extra;"
>

<!ENTITY % InlNoAnchor.class
     "%InlStruct.class;
      %InlPhras.class;
      %InlPres.class;
      %InlSpecial.class;
      %InlForm.class;
      %Inline.extra;"
>

<!ENTITY % InlNoAnchor.mix
     "%InlNoAnchor.class;
      %Misc.class;"
>

<!ENTITY % Inline.mix
     "%Inline.class;
      %Misc.class;"
>

<!-- Block Elements  ............................................ -->

<!ENTITY % Heading.class
     "%h1.qname; | %h2.qname; | %h3.qname;
      | %h4.qname; | %h5.qname; | %h6.qname;"
>
<!ENTITY % List.class  "%ul.qname; | %ol.qname; | %dl.qname;" >

<!ENTITY % Table.class "| %table.qname;" >

<!ENTITY % Form.class  "| %form.qname;" >

<!ENTITY % BlkStruct.class "%p.qname; | %div.qname;" >

<!ENTITY % BlkPhras.class
     "| %pre.qname; | %blockquote.qname; | %address.qname;"
>

<!ENTITY % BlkPres.class "| %hr.qname;" >

<!ENTITY % BlkSpecial.class
     "%Table.class;
```

```
      %Form.class;"
>

<!ENTITY % Block.extra "" >

<!ENTITY % Block.class
     "%BlkStruct.class;
      %BlkPhras.class;
      %BlkPres.class;
      %BlkSpecial.class;
      %Block.extra;"
>

<!ENTITY % Block.mix
     "%Heading.class;
      | %List.class;
      | %Block.class;
      %Misc.class;"
>

<!-- All Content Elements  ...................................... -->

<!ENTITY % FlowNoTable.mix
     "%Heading.class;
      | %List.class;
      | %BlkStruct.class;
      %BlkPhras.class;
      %BlkPres.class;
      %Form.class;
      %Block.extra;
      | %Inline.class;
      %Misc.class;"
>

<!ENTITY % Flow.mix
     "%Heading.class;
      | %List.class;
      | %Block.class;
      | %Inline.class;
      %Misc.class;"
>

<!-- end of XHTML-Print 1.0 Document Model Module  ........................ -->
<!-- ...................................................................... -->
```