A Technical Introduction to PDF/UA

PDFlib *Whitepaper*

The PDF/UA Standard for Universal Accessibility

In the early days PDF documents didn't have a good reputation regarding basic accessibility principles. While PDF does a great job at faithfully preserving the visual appearance of a document, the logical structure of the content is generally not preserved in PDF, mostly because the format is rooted in PostScript printing technology and not in structured document formats such as XML. This shortcoming has been addressed with the introduction of Tagged PDF in PDF 1.4 (Acrobat 5, released in 2001), but adoption has been slow during the first ten years. The PDF/UA standard ISO 14289, published in 2012 and slightly updated in 2014, is built on these structured document capabilities and further expands Tagged PDF. In the words of the standard:

»The primary purpose of ISO 14289 (known as PDF/UA) is to define how to represent electronic documents in the PDF format in a manner that allows the file to be accessible.«

Web Content Accessibility Guidelines (WCAG) 2.0

In order to understand accessibility requirements for PDF it is useful to take a look at accessibility requirements for the Web. The W3C published its »Web Content Accessibility Guidelines« WCAG 2.0 in 2008 (also standardized as ISO/IEC 40500:2012). It contains twelve general accessibility guidelines which are organized according to the following principles:

- Content should be perceivable, e.g. alternate text should be available for images and other non-textual content.
- Content should be operable, e.g. all functionality should be available from a keyboard.
- Content should be understandable, e.g. all text should be readable.
- Content should be robust for compatibility with current and future tools.

These guidelines describe a total of 61 success criteria which must be met to make content accessible. The success criteria are grouped in three conformance levels, ranging from A (minimum conformance) to the medium level AA and up to the highest level AAA. In order to achieve minimum accessibility requirements only Level A criteria must be met.

Applying WCAG to PDF

WCAG is a technology-independent standard: it is not targeted at a specific technology such as HTML or server-side scripting, but describes accessibility requirements in a very general manner. An advisory (i.e. non-normative) document called »Techniques for WCAG 2.0« discusses general techniques for achieving WCAG conformance and provides techniques for specific technologies such as HTML, CSS, Flash and PDF. The advisory »PDF Techniques for WCAG 2.0« applies WCAG requirements to PDF and provides useful examples which demonstrate in detail how a particular accessibility requirement can be met. The examples are targeted at users of Acrobat, content creation software (e.g. Microsoft Word, OpenOffice) and PDF developers.

For the convenience of PDFlib users the document »PDFlib Techniques for WCAG 2.0« (available on the PDFlib Web site) explains procedures which can be used to meet WCAG requirements when creating accessible documents with PDFlib.

WCAG and PDF/UA

While the W3C's PDF techniques are useful, they do not cover all technical PDF details for achieving full accessibility. The PDF/UA standard clarifies and simplifies PDF requirements to meet WCAG 2.0. While PDF/UA conformance ensures WCAG conformance for native PDF page content, there are some areas where WCAG imposes additional requirements. In particular, WCAG requirements must be met in addition to PDF/UA for embedded multimedia content and scripting in order to make such content types in PDF accessible.

The relationship of WCAG and PDF/UA rules is explained in detail in a document published by AIIM. »Achieving WCAG 2.0 with PDF/UA« contains a reference table which maps WCAG success criteria to the corresponding PDF/UA requirements.

There are only two topics where PDF/UA imposes additional requirements beyond WCAG: in order to improve document navigation PDF/UA mandates various nesting rules for heading tags, while WCAG isn't concerned about nesting of headings. PDF article threads must be applied appropriately; however, this PDF feature is somewhat outdated and rarely used in newly created PDF documents.

Note that a PDF document may conform to WCAG 2.0 without conforming to PDF/UA, but achieving WCAG conformance is much simpler with PDF/UA than without.

Tagged PDF Basics

Since PDF/UA is based on Tagged PDF let's take a look at this first. While PDF initially has been designed to faithfully preserve the graphical appearance of a document, it doesn't necessarily contain information about a document's structure. For example, section headings may be printed in large or bold type, but there is no explicit »heading« marker available in plain PDF documents. Similar to XML-based markup languages, content in Tagged PDF can be »marked« and incorporated in a structural document hierarchy. Each relevant content item has a designated place in this hierarchy. Content items which don't contribute to the vital document contents (e.g. page numbers) are marked as *Artifact*.

The logical structure in a Tagged PDF document is described by a hierarchy of elements, called the structure hierarchy (or logical structure tree, or tag tree). Starting at the root level (often called the *Document* element), the structure hierarchy consists of an arbitrary number of levels. On each level an element may contain zero or more items of the following kinds:

- Other structure elements, e.g. the *Document* element may contain multiple *Article* elements, and each *Article* element may in turn contain multiple *P* (paragraph) elements.
- Content items such as marked sequences of text and graphics on the page, XObjects created from imported images or PDF pages, or annotations and form fields. These items represent the content associated with a structure element.

Tagged PDF solves the potential conflict between content creation order and logical reading order: the contents of a PDF page can be placed in any order, but reading text in this native order does not necessarily match the logical order of page contents. In contrast, the logical structure tree arranges page contents according to their logical order, i.e. the order in which a human expects to read it.

PDF/UA-1 File Format Requirements

PDF/UA-1 is based on ISO 32000-1 (PDF 1.7). It doesn't add any new features to the PDF file format, but makes some aspects required which are optional in PDF 1.7. The following conditions must be met in all PDF/UA-1 documents:

- The document must be tagged. While PDF 1.7 includes some requirements regarding the nesting and relationship of different types of structure elements, PDF/UA-1 extends and clarifies these rules (see below for details).
- All fonts used in the document must be embedded (except fonts for invisible text, e.g. OCR results).
- Some layer options are not allowed.
- Reference XObjects for external page content (as used in PDF/X-5) are not allowed.
- The document title must be specified in the document's metadata.

Semantic requirements

When creating the structure hierarchy for PDF/UA-1, the following semantic aspects must be obeyed:

- Tagging must use structure elements which are appropriate for the document structure: if it's a heading, it must be tagged as heading. If it's a table, it must be tagged as a table. If it's a list, it must be tagged as a list.
- Contents which are not relevant for the document's meaning must not be included in the document hierarchy, but must instead be tagged as *Artifact*. Typical examples are running headers and footers, page numbers, and background images.
- Structure elements must be arranged in logical reading order.
- Content must be tagged appropriately if the intended information is not otherwise accessible because of the content's color, format or layout.
- Text represented in a graphic requires the *Alt* attribute with an explanation if it doesn't contain text in a natural language (e.g. font or script samples).
- Images must provide alternative text; image captions must be marked with a *Caption* tag.
- Links must be accompanied by a suitable *Link* annotation.
- Only a single *Figure* tag must be created for groups of graphical elements which logically belong together.
- Footnotes, endnotes, note labels and references to locations within the document must be tagged as *Note* or *Reference* as appropriate.

Because of the semantic requirements outlined above it is difficult or impossible to automatically convert existing untagged PDF documents to conforming PDF/UA. Similarly, applying OCR techniques to scanned documents is unlikely to result in fully conforming PDF/UA without human intervention. For example, alternative text for images cannot be derived automatically.

Requirements for specific tags

All standard tags defined in PDF 1.7 may be used in PDF/UA-1. If other tags are used, a mapping of those custom tags to the standard tags must be provided in the document's *Rolemap*. Various rules must be obeyed regarding certain standard element types:

- The *Figure* element used for images and graphics which are not artifacts requires an *Alt* or *ActualText* attribute.
- *Table* elements must be created for logical tables, but must not be created for tables which are created for layout purposes. Table-related tags must be properly nested, e.g. *Table* tag contains table rows *TR* which in turn contains table header cells *TH* or table data cells *TD*. The *Scope* attribute is required for *TH* (table header) elements.
- Heading tags must be properly nested. If numbered heading tags *(H1, H2, ...)* are used these must be properly nested (i.e. levels must not be skipped). If unnumbered heading tags are used (so-called strongly structured documents) the *H* tag must be used, but not more than once in each node of the structure tree. Heading elements must not have any descendants.
- The list element type *L* requires a *ListNumbering* attribute which designates the numbering system used in the list, e.g. *Disc* for a simple bullet without numbers, *Decimal*, or *UpperRoman*.

Requirements for specific content types

The following requirements must be met for various types of PDF content:

- The natural language of text must declared, either with the *Lang* document info entry for the whole document or with the *Lang* attribute of individual structure elements. Invisible text must be tagged as *Artifact* unless it has a rendered equivalent (e.g. a scanned image).
- Vector graphics and raster images must be tagged as *Figure* or *Artifact*.
- Annotations and form fields must be included in the structure tree and require certain flags to ensure accessibility.

Unicode requirements

PDF/UA requires proper Unicode semantics for all text in the document. This requirement is rooted in the fact that PDF supports a variety of font and encoding techniques, not all of which support Unicode. For example, PDF supports PostScript Type 1 fonts which have been introduced in the 1980's, while the Unicode consortium started its work in 1991. PDF/UA requires that supplementary Unicode mapping information must be present for fonts which do not contain it internally. But not all Unicode values are acceptable: values in the Private Use Area (PUA) are not allowed since they do not carry any common interpretation (semantics).

Symbolic fonts are an important area where this PDF/UA requirement holds, e.g. fonts containing logos or pictograms. Since standardized Unicode values are not available for custom symbolic glyphs, suitable

Unicode semantics must be provided in an *ActualText* attribute. The *ActualText* may be assigned to an individual glyph or a sequence of multiple glyphs, and may contain an arbitrary Unicode string.

As an example, code 0x1A in the common WingDings font contains an image of a computer keyboard with the glyph name *keyboard* and the Unicode value U+F037 in the PUA range, which is not acceptable in PDF/UA-1. For lack of better substitute text the glyph name could be used to construct suitable *ActualText*, e.g. »symbol for keyboard«. It should be noted that programmatically constructing ActualText must be considered a makeshift solution; human-selected text is always preferable to machine-generated ActualText.

Other recommendations

While not strictly required in PDF/UA-1, the following items are recommended:

- Bookmarks are recommended for improved navigation. They should reflect proper reading order and nesting of the content.
- Tables should include headers.
- Attachments should be accompanied by a description, and should be accessible in their own right.
- If present, page labels (e.g. roman page numbers) should be appropriate.

PDF/UA and PDF/A

The archiving standards PDF/A-1a, PDF/A-2a and PDF/A-3a require the use of Tagged PDF. Although there is no direct relationship between PDF/A-1a/2a/3a and PDF/UA-1, a document can conform to both standards at the same time. In fact, if you want to create PDF/A with conformance level A we recommend to adhere to the PDF/UA-1 requirements as well in order to improve accessibility. For more information please refer to the PDF/A Whitepaper on the PDFlib Web site.

We recommend to avoid PDF/A-1a and work with the newer PDF/A-2a or PDF/A-3a standards instead because there is a minor conflict between PDF/UA-1 and PDF/A-1a: PDF/UA-1 requires the *Tabs* entry for pages with annotations. This key specifies the tab order for the page's annotation and must specify »structure order«. However, this key is not available in PDF 1.4 and thus cannot be used in combined PDF/A-1a and PDF/UA-1 documents.

The Matterhorn Protocol

The PDF/UA-1 standard defines rules to which a document must obey in order to conform to the standard. The PDF/UA Competence Center, an initiative of the PDF Association, compiled a comprehensive list of all possible PDF/UA-1 standard violations. This list, called the Matterhorn Protocol, provides guidance to software developers and document testers for achieving full accessibility.

The Matterhorn Protocol rephrases the standard conformance requirements and lists all possible violations as 136 »failure conditions« which are grouped in 31 checkpoints. While most failure conditions can be identified by software, others require human judgment. Some examples for failure conditions which can be identified by software:

- An image is missing alternative or replacement text.
- A link annotation is not nested within a *Link* tag.
- XMP metadata does not include a document title.

Some examples for failure conditions which require human judgment:

- Heading text is not tagged with a heading tag.
- Irrelevant content is not tagged as *Artifact*.
- OCR'ed text contains character identification errors.
- Content is presented as a list, but is not tagged as a list.

The Matterhorn Protocol can be used as a guideline for identifying PDF/UA-1 standard violations. For example, the PDF Accessibility Checker (PAC) is a PDF/UA validation tool which is freely available from the Swiss non-profit organization Access for All. PAC implements the Matterhorn Protocol.

PDF/UA conforming Readers and Assistive Technology (AT)

The PDF/UA-1 standard not only specifies file format requirements, but also the behavior of a »conforming reader« and »assistive technology« (AT). A conforming reader is a PDF consumer according to ISO 32000-1 (the PDF 1.7 standard) which adheres to additional provisions. AT is »software and/or hardware used by a person with a disability that provides alternative controls and/or renditions to facilitate their access to and usage of available functionality and information«. AT doesn't usually interpret PDF documents directly, but in combination with a PDF reader. This may be done via interoperability interfaces such as MSAA on Windows. Common examples of AT are a screen reader program which presents text on the screen on a Braille device, and speech synthesis software which reads the document contents to the user. A conforming reader must correctly interpret accessibility-related information in a PDF/UA document and provide it to AT. It must support document navigation via page labels, structure hierarchy or bookmarks, and cooperate with AT by making available the following items:

- structure elements and artifacts;
- its user interface;
- text in logical reading order along with the natural language of the text;
- information about empty table cells;
- information about the presence and names of layers (optional content);
- names and contents of file attachments;
- metadata and digital signature information;
- alternate description of annotations and descriptions of form fields

A conforming AT device must consume and present the information provided by the conforming reader. It must also process artifacts, i.e. document content which is not included in the structure tree. Like a conforming reader, conforming AT must support document navigation via page labels, structure hierarchy or bookmarks.

Commercially available assistive software is expected to support the PDF/UA-1 standard in the future. The NVDA project offers a free, open source screen reader for Windows. NVDA already announced their plans to incorporate PDF/UA support in their software.

PDF/UA Support in PDFlib GmbH Products

All products in the PDFlib 9 family support the creation of PDF/UA according to the standard:

- The PDFlib base product can be used to create PDF documents based on text, graphics and images along with the corresponding structure tags. Tagged PDF functionality in PDFlib has already been introduced in 2004.
- The extended product PDFlib+PDI additionally allows the import of pages from PDF/UA documents including the associated part of the structure hierarchy. This makes PDFlib+PDI 9 the first product worldwide which can be used to assemble existing PDF/UA documents to combined output which in turn conforms to the standard (sometimes called »content aggregation«).
- PDFlib Personalization Server (PPS) is the top-level product for template-based PDF creation. It also supports PDF/UA output which means that personalized accessible PDF documents can be created.

PDFlib is not simply a low-level toolkit which makes it possible to somehow create PDF/UA-1. Instead, PDFlib implements all rules and requirements of the PDF/UA standard: it automatically creates accessible constructs where possible, and prevents the user from violating any PDF/UA requirement. The resulting output is guaranteed to conform to all PDF/UA requirements as far as machine-testable criteria are concerned. Criteria which require human judgment are obviously under the user's responsibility.

PDFlib's support for PDF/A-1a, PDF/A-2a and PDF/A-3a in combination with PDF/UA-1 makes it easy to create PDF output which is both accessible and archivable.

Creating PDF/UA with PDFlib

PDFlib 9 takes care of PDF/UA-1 requirements automatically as far as possible. Developers working with PDFlib products must provide structure information to the PDFlib programming interface so that PDFlib can create the document's structure hierarchy. PDFlib applies PDF/UA-1 checks to all user-supplied structure tags and attributes to ensure conforming output. PDFlib automatically tags tables and artifacts which is a big time-saver for the developer and simplifies the creation of a standard-conforming structure hierarchy. Alternative text can be attached to images and vector graphics.

Sample code for a variety of programming languages and development environments is provided with the PDFlib distribution. Additional programming techniques for PDF/UA are available in the PDFlib Cookbook.

Assembling PDF/UA with PDFlib+PDI and PPS

PDFlib+PDI and the PDFlib Personalization Server (PPS) can be used to assemble pages from existing documents and enhance them with new content. PDFlib+PDI 9 introduced the capability to import the corresponding parts of the structure hierarchy with each page and merge it into the generated document's structure hierarchy. If the imported document conforms to PDF/UA, the resulting output also conforms to the standard. With this functionality accessible compound documents can be assembled from smaller parts.

Digitally signing PDF/UA with PLOP DS

PDFlib PLOP DS is our product for applying digital signatures to PDF documents. PLOP DS applies signatures in a PDF/UA-conforming manner: if the input conforms to PDF/UA, the signed output is guaranteed to conform to the PDF/UA standard as well.

PDFlib GmbH
Franziska-Bilek-Weg 9
80339 München, Germany
phone +49 • 89 • 452 33 84-0
support@pdflib.com
www.pdflib.com/knowledge-base/pdfua

PDFlib GmbH is completely focused on PDF technology. Customers worldwide use PDFlib products since 1997. The company closely follows development and market trends, such as ISO standards for PDF. PDFlib GmbH products are distributed all over the world with major markets in North America, Europe, and Japan.

Founded in 2006 as the PDF/A Competence Center, the PDF Association exists to promote the adoption and implementation of International Standards for PDF technology.

- Developers use the PDF Association to share knowledge and experience with PDF technology.
- Decision-makers use the PDF Association to learn about the role and capabilities of PDF and PDF's subset standards in ECM and other electronic document applications.
- End-users benefit from improved reliability, quality and functionality and interoperability in their experience of electronic documents.