Tagged PDF and PDF/A compliance

Tagged PDF

Tagged PDF files contain information about the structure of the document. The information about the structure is transported via so-called "PDF tags". Tagging a PDF makes it more accessible to screen readers, handhelds and similar devices. Enabled tagging also improves the copy and paste behavior. For example, copying a whole paragraph in a tagged PDF created with PDFreactor will ignore the line breaks which are displayed in the PDF document. Furthermore tagging applies reflow.

Using the `setAddTags` API method, you can add PDF tags to the PDF documents generated with PDFreactor. If you are generating a PDF from HTML documents, the HTML elements are automatically mapped to the corresponding PDF tags, so all you have to do is setting this property to enable tagging.

The following example maps the HTML element image to the PDF tag "Figure", and the content of its alt attribute to an alternative description for this tag.

```
img {
    -ro-pdf-tag-type: "Figure";
}
img[alt] {
    -ro-alt-text: -ro-attr(alt);
}
```

The screenshot (taken from Adobe Acrobat DC) on the right shows that PDFreactor is capable to tag even complex structures such as tables properly. The table below was placed on the bottom of the page to demonstrate that PDFreactor wont repeat the <Table> or <THead> tag even though the table splits onto another page.

A tagged PDF will often be bigger then an equivalent PDF file that does not include PDF tags. You can enable the full compression mode to reduce the document size. To do so, the method setFullCompression can be used in the PDFreactor integration:

```
config.setFullCompression(true);
```

Example table

Employee	Mail	Phone
John Doe	johne.doe@example.com	202-555-0152
Austin King	austin.king@example.com	202-555-0191
Edward Alsop	edward.alsop@example.com	202-555-0113
Brian Mitchell	brian.mitchell@example.com	202-555-0131

PDF/A Conformance

PDF/A differs from PDF by prohibiting features ill-suited to long-term archiving, such as font linking (as opposed to font embedding).

The PDF/A standard does not define an archiving strategy or the goals of an archiving system. It identifies a "profile" for electronic documents that ensures the documents can be reproduced exactly the same way using various software in years to come. A key element to this reproducibility is the requirement for PDF/A documents to be 100% self-contained. All of the information necessary for displaying the document in the same manner is embedded in the file. This includes, but is not limited to, all content (text, raster images and vector graphics), fonts and color information. A PDF/A document is not permitted to be reliant on information from external sources (e.g. font programs and data streams), but may include annotations (e.g. hypertext links) that link to external documents.

PDFreactor supports the creation of all PDF/A conformant files.

Many companies and government organizations worldwide require PDF/A conformant documents.

PDF/A-1a is the most strict PDF/A standard while the newer PDF/A standards are more lenient, e.g. allowing transparency and attachments.

Common PDF/A conformance requirements

PDF/A restriction	PDFreactor actions
All used fonts are embedded.	PDFreactor ignores the option to disable font embedding.
All images are embedded.	Images are always automatically embedded by PDFreactor.
Multi-media content is prohibited.	Embedding objects is automatically prevented by PDFreactor, when PDF/A conformance is set.
JavaScript is prohibited.	No JavaScript is embedded, when PDF/A conformance is set. (This does not prohibit JavaScript in the source HTML document to processed during conversions)
Encryption is disallowed.	This is automatically prevented, when the PDF/A conformance is set.
The PDF must be tagged.	This is automatically done by PDFreactor, when PDF/A conformance is set.
Metadata included in the PDF is required to	This is automatically done by PDFreactor, when PDF/A conformance is set.

PDF/A restriction	PDFreactor actions
be standard-based XMP.	
Colors are specified in a device-independent manner.	In PDFreactor colors are defined either as RGB or CMYK. When PDF/A conformance is set, one of these color spaces has to be set in conjunction with a color space profile. CMYK requires an ICC profile to be set, RGB colors use a default sRGB profile, if no other is set. Using RGB colors in CMYK PDF/A documents or vice versa is prohibited. Color keywords and shades specified via the "gray" function are converted to the appropriate color space losslessly.

PDF/A-1a specific conformance requirements

PDF/A-1a restriction	PDFreactor actions
Transparency is disallowed.	PDFreactor will ignore certain kinds of transparency of images. Other occurrences of transparency will cause an exception to be thrown.
Attachments are disallowed.	This is automatically prevented, when PDF/A-1a conformance is set.

To create a PDF/A conformant document, the method setConformance can be used in the PDFreactor integration:

```
config.setConformance(Conformance.PDFA3A);
```

If CMYK colors are used in a document to be converted into a PDF/A-conformant file, an Output Intent has to be set. It is possible to use the following API methods:

```
Configuration config = new Configuration();

OutputIntent outputIntent = new OutputIntent();
outputIntent.setIdentifier("ICC profile identifier");

// Use this if you are loading the ICC profile via URL
outputIntent.setUrl("URL/to/ICC/profile");

// Use this if you want to specify the ICC profile's binary data
outputIntent.setData("ICC profile binary data");

config.setOutputIntent(outputIntent);
```

The first parameter is a string identifying the intended output device or production condition in human- or machine-readable form. The second parameter points to ICC profile file or contains data of such a profile.

Note:
When PDF/A conformance is set, encryption, restrictions, comments, full compression and other non PDF/A-conformant features are automatically overwritten, regardless of their own settings.
Setting PDF/A-1a conformance generates PDFs with Adobe PDF version 1.4 in which some PDF tags are forbidden e.g. <TBody>. PDFreactor will skip all forbidden tags automatically, but handle table headers correctly.