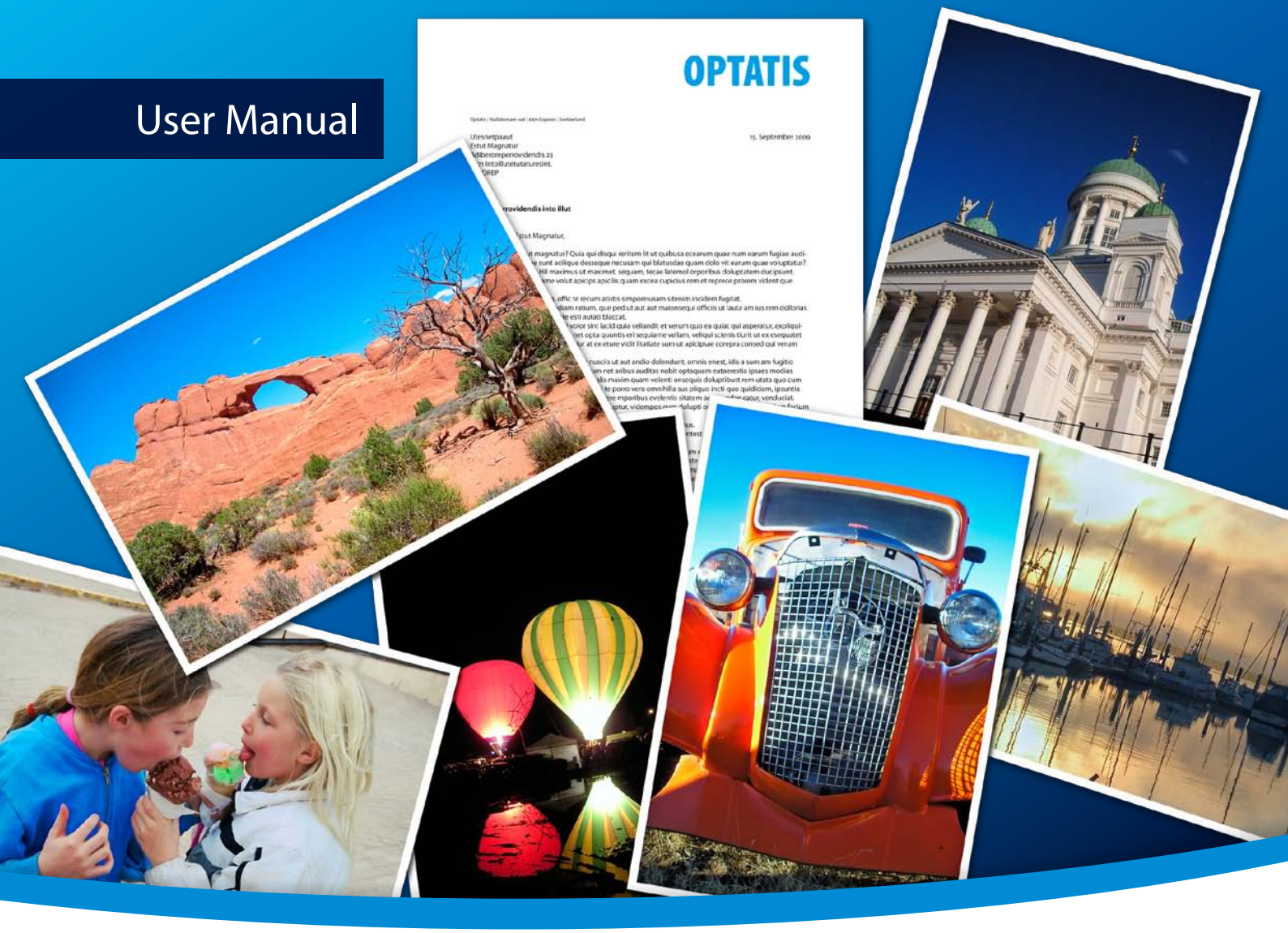
User Manual



3-Heights™ TIFF Toolbox Shell

PDF/A conform

Version 6.14.1

PDF-TOOLS.COM
Premium PDF Technology

Contents

1 Introduction

1.1 Description

The 3-Heights™ TIFF Toolbox Shell enables to process and convert TIFF files. Examples of such processing functions are the merging and splitting of multi-page files, the conversion of other raster image formats such as JPEG into TIFF and the conversion of TIFF files into PDF/A. The optional OCR add-in makes output files searchable in full text mode.

PDF/A has been acknowledged world-wide as the ISO standard for long-term archiving since 2005. The TIFF to PDF/A converter is used to convert images into a standardized format, for instance for electronic archiving or electronic data exchange.

The 3-Heights™ TIFF Toolbox Shell is characterized by a robust design, high throughput, ability to process large files and accurate image reproduction.

1.2 Functions

The 3-Heights™ TIFF Toolbox Shell consists of the following shell tools

`tiff2pdf` Convert a multi-page TIFF file into a PDF/A-1 or PDF/A-2 file.

`tiffcompress` Re-compress a multi-page TIFF file using the selected algorithms. Split each page of a multi-page TIFF file with the Mixed Raster Content (MRC) technique into foreground, background and mask layer, compress these layers, sample-down the foreground and background.

`tiffimport` Import raster image formats such as JPEG into a TIFF file.

`tiffmerge` Merge single- or multi-page TIFF files into one multi-page TIFF file.

`tiffocr` Perform optical character recognition using the optional OCR add-in, calculate mask or detect photographic pictures. Embed the recognized text in the resulting TIFF file.

`tiffsplit` Split a multi-page TIFF file into single-page TIFF files.

`tiffextract` Extract ICC color profiles, OCR data or XMP metadata from TIFF.

`tiffscan` Scan image from a TWAIN or WIA scanner into one multi-page file.

1.2.1 Features

- Support for mixed raster content (MRC) layers during compression and conversion to PDF/A.
- Support for optical character recognition (OCR) and searchable PDFs through optional OCR add-in.
- Support for recognition of photographic picture regions.
- The JPEG import supports the RGB and CMYK color spaces of the JFIF and the Adobe Photoshop file formats.
- Support for "old-JPEG" according to TIFF V6 and "new-JPEG" according to Tech. Note #2 including the conversion from old to new format.
- Compression support for JPEG2000 and JBIG2 in addition to the TIFF V6 algorithms.
- Support for ICC color profiles.
- Support for XMP metadata.
- Substitution of a default resolution where missing.

- Support for merging and splitting of multipage TIFFs.
- Support for extracting ICC color profiles and XMP metadata from TIFF.
- Support for PDF/A user units to handle large TIFF page dimensions.
- Return codes allow for error handling in shell scripts.
- Support for scanning with scanners that provide the TWAIN or WIA interface.

1.2.2 Formats

Input Formats

- JPEG
- TIFF V6 and Tech. Note #2

Output formats

- TIFF V6 and Tech. Note #2
- PDF 1.x (PDF 1.0, . . ., PDF 1.7)
- PDF 2.0
- PDF/A-1, PDF/A-2, PDF/A-3

1.2.3 Conformance

Standards

- ISO 32000-1 (PDF 1.7)
- ISO 32000-2 (PDF 2.0)
- ISO 19005-1 (PDF/A-1)
- ISO 19005-2 (PDF/A-2)
- ISO 19005-3 (PDF/A-3)
- TIFF V6 (http://www.adobe.com)
- Tech. Note #2 (http://www.ijg.org)
- RFC 2301 (https://www.ietf.org/)

1.3 Operating Systems

The 3-Heights™ TIFF Toolbox Shell is available for the following operating systems:

- Windows Client 7+ | x86 and x64
- Windows Server 2008, 2008 R2, 2012, 2012 R2, 2016, 2019 | x86 and x64
- Linux:
  - Red Hat, CentOS, Oracle Linux 7+ | x64
  - Fedora 29+ | x64
  - Debian 8+ | x64
  - Other: Linux kernel 2.6+, GCC toolset 4.8+ | x64
- macOS 10.10+ | x64

'+' indicates the minimum supported version.

2 Installation

2.1 Windows

The 3-Heights™ TIFF Toolbox Shell comes as a ZIP archive or as an MSI installer.

The installation of the software requires the following steps.

1. You need administrator rights to install this software.
2. Log in to your download account at http://www.pdf-tools.com. Select the product "TIFF Toolbox Shell". If you have no active downloads available or cannot log in, please contact pdfsales@pdf-tools.com for assistance.
   You will find different versions of the product available. We suggest to download the version, which is selected by default. A different version can be selected using the combo box.
   There is an MSI (`*.msi`) package and a ZIP (`*.zip`) archive available. The MSI (Microsoft Installer) package provides an installation routine that installs and uninstalls the product for you. The ZIP archive allows you to select and install everything manually.
   There is a 32 and a 64-bit version of the product available. While the 32-bit version runs on both, 32 and 64-bit platforms, the 64-bit version runs on 64-bit platforms only. The MSI installs the 64-bit version, whereas the ZIP archive contains both the 32-bit and the 64-bit version of the product. Therefore, on 32-bit systems, the ZIP archive must be used.
3. If you select an MSI package, start it and follow the steps in the installation routine.
4. If you are using the ZIP archive, do the following. Unzip the archive to a local folder, e.g. `C:\Program Files\PDF Tools AG\`.
   This creates the following subdirectories:

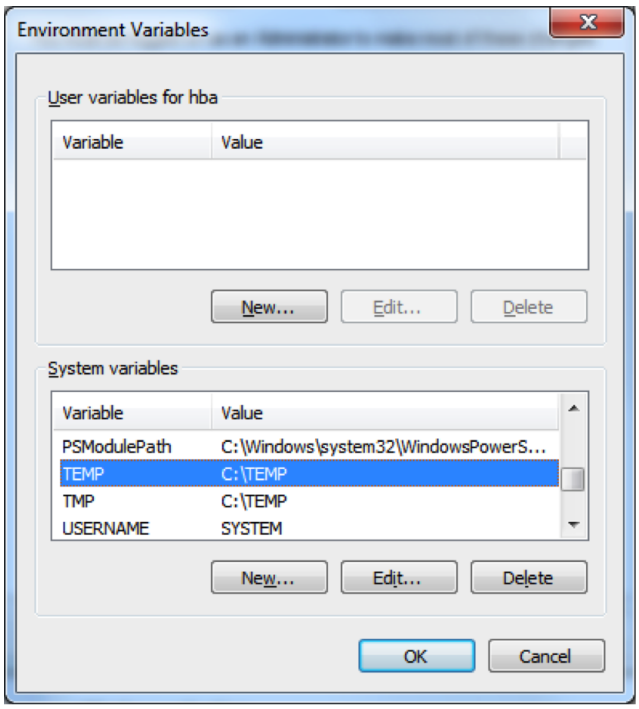
Subdirectory	Description
`bin`	Contains the runtime executable binaries.
`doc`	Contains documentation.

5. (Optional) To easily use the 3-Heights™ TIFF Toolbox Shell from a shell, the directory needs to be included in the "Path" environment variable.
6. (Optional) Register your license key using the License Management.

2.1.1 How to set the Environment Variable "Path"

To set the environment variable "Path" on Windows, go to Start → Control Panel (classic view) → System → Advanced → Environment Variables.

Select "Path" and "Edit", then add the directory where `tiff2pdf.exe` is located to the "Path" variable. If the environment variable "Path" does not exist, create it.



2.2 Unix

2.2.1 32 bit Version

Unpack the archive in an installation directory, e.g. `/usr/local`.

Include the bin directory in the $PATH environment variable. The commands for the various platforms are:

Linux, Solaris, HP-UX, AIX `export PATH=\$PATH:/usr/local/bin:.`

macOS `setenv PATH \$PATH:/usr/local/bin:.`

2.2.2 64 bit Version

The 64 bit versions are dynamically linked and require a shared library. The performance of the 64 bit and the 32 bit version is equal, it is suggested to use the 32 bit version, unless the application requires otherwise. The shared libraries can be downloaded from the web site at https://www.pdf-tools.com.

The shared library needs to be copied to the `lib` sub-directory, which needs to be included in the appropriate environment variable that is used by the operating specific loader (`ld`).

The shell commands are (depending on the command shell):

Linux, Solaris, HP-UX `export LD\_LIBRARY\_PATH=/usr/local/lib`

AIX `export LIBPATH=/usr/local/lib`

macOS `setenv DYLD\_LIBRARY\_PATH /usr/local/lib`

2.3 Uninstall

If you have used the MSI for the installation, go to Start → 3-Heights™ TIFF Toolbox Shell. . . → Uninstall . . .

If you have used the ZIP file for the installation: In order to uninstall the product, undo all the steps done during installation.

2.4 Note about the Evaluation License

With the evaluation license the 3-Heights™ TIFF Toolbox Shell automatically adds a watermark to the output files.

3 License Management

The 3-Heights™ TIFF Toolbox Shell requires a valid license in order to run correctly. If no license key is set or the license is not valid, then the executable will fail and the return code will be set to 10.

More information about license management is available in the license key technote.

4 Getting Started

4.1 Invoking the Commands

The simplest command requires an input image file parameter and an output PDF file as parameters:

```
tiff2pdf input.tif output.pdf
```

It converts a TIFF file to a PDF/A-2 document. If the image is multi-page TIFF image, then each page in the image will be converted to a page in the PDF output document.

To concatenate several TIFF files into one TIFF file, add the input files as additional parameters before the output file:

```
tiffmerge input1.tif other*.tif output.pdf
```

> **Note:** You can use any number of input files including wildcards.

To split a TIFF file into single pages you can use the command:

```
tiffsplit input.tif out-%03d.pdf
```

The sub-string "%03d" is replaced with a three-digit page number with leading zeros.

4.2 Specify the Folder of the Output File

The output folder can simply be added in front of the output file name

```
tiff2pdf input.pdf myfolder\output.pdf
```

or absolute (Windows):

```
tiff2pdf input.pdf C:\myfolder\output.pdf
```

4.3 The Use of Wildcards (*.)

The 3-Heights™ TIFF Toolbox Shell supports wildcards. If a directory for example contains the following input JPEG files:

```
A01.jpg
A02.jpg
A03.jpg
B01.jpg
B02.jpg
```

Then the following command processes all JPEG files starting with the letter "A".

```
tiff2pdf A*.jpg output.pdf
```

Note: The file extension of the input files must always be a supported format. When using wildcards, it is helpful to set the verbose mode option `-v`. The command then looks like this:

```
tiff2pdf -v *.jpg output.pdf
```

And the generated output message looks like this:

```
Converting file A01.jpg (1)
Converting file A02.jpg (2)
...
Done.
```

Wildcards return a list of existing files. If you would like to convert all files in a directory to individual output files, it is required to use a variable to name the output files.

Example: Use the `for` command of the Windows CMD shell, to convert all JPEG files to individual PDF files with the same name and the extension `.pdf`, in the same directory:

```
for %f in (*.jpg) do tiff2pdf -v %f %~nf.pdf
```

Example: Of course, one can adjust the paths, or use a different output name:

```
for %f in (C:\InputDir\*.jpg) do tiff2pdf -v %f C:\OutputDir\%~nf.pdf
```

Note: Variables used in a batch file (`.bat`) require two leading `%` instead of one.

4.4 Verbose vs. Quiet Mode

When using wildcards, it might be helpful to set the verbose mode option. The command then looks like this:

```
tiff2pdf -v *.tif output.pdf
```

And the generated output message looks like this:

```
- Output file c:\test\output.pdf created.
- Input file c:\test\A01.tif opened.
- Input file c:\test\A02.tif opened.
- Input file c:\test\A03.tif opened.
Done.
```

The output messages can be redirected to a file using the redirection operator:

```
tiff2pdf -v *.tif output.pdf >log.txt
```

4.5 Processing Files in a Directory

Wildcards return a list of existing files. If you would like to convert all files in a directory to individual PDF documents, it is required to use a variable to name the output files. Here is an example for the `for`-command of the CMD-shell. It converts all TIFF files to PDFs with the same name and the extension `.pdf`, in the same folder:

```
for %i in (*.tif) do tiff2pdf -v %i %~ni.pdf
```

Of course, one can adjust the paths, or use a different output name:

```
for %i in (.\input\*.tif) do tiff2pdf %i .\output\new_%~ni.pdf
```

For additional help to the `for`-command, use the command:

```
for /?
```

> **Note:** Variables used in a batch file require two leading `%` instead of one.

4.6 Mixed Raster Content (MRC)

Some raster images—typically scanned documents—consist mainly of text, possibly in several colors and interspersed with some pictures. Such images are difficult to compress with one single compression type because of the diverse or even conflicting features of different parts of the image.

The MRC technique is a way of breaking such images down into parts, such that each part is well suited for one type of a compression algorithm. With this approach, the resulting file size often can be reduced without significantly reducing the visual quality of the document.

> **Note:** MRC optimization can only be enabled for continuous images, i.e. not for bi-tonal images and images with an indexed color space.

4.6.1 Examples

Conversion of a TIFF into a PDF/A-2B, perform MRC, downsample foreground and background layer and compress them with JPEG2000.

```
tiffocr -ocr internal input.tif outOcr.tif
tiffcompress -s -cm 34712 -qm 10 -dm 3 outOcr.tif outComp.tif
tiff2pdf -cl pdfa-2b outComp.tif out.pdf
```

Conversion of a TIFF into a PDF/A-1B, recognize photographic pictures, perform MRC, downsample foreground and background layer and compress them with JPEG.

```
tiffocr -ocr internal -ocp "RecognizePictures=true" input.tif outOcr.tif
tiffcompress -s -cm 6 -qm 30 -dm 4 outOcr.tif outComp.tif
tiff2pdf -cl pdfa-1b outComp.tif out.pdf
```

4.6.2 Phase 1: Recognizing Photographic Pictures

In this phase, the `tiffocr` tool computes a bi-tonal mask. Optionally rectangular areas containing photographic features are detected. The mask and the information about the location of the detected photographic pictures is embedded in the resulting TIFF.

It is possible, that actual photographic regions present in the input image are not recognized correctly. This can happen for example if a photographic region contains parts with uniform color.

4.6.3 Phase 2: Segmentation into Layers

In this phase the `tiffcompress` tool is used.

All photographic pictures that have been recognized in phase 1 are removed from the input image and are embedded as separate images in the resulting TIFF file. Thereafter the image is supposed not to contain photographic features anymore. Instead, the image is assumed to consist of text and graphic, potentially with varying color.

Now, using the mask from phase 1 the whole image is separated into three layers, a foreground, a background and a mask layer. The mask, which can be thought of as a bi-tonal image tells for each pixel whether to show the corresponding pixel of the foreground layer or the background layer.

Example:

Let the image consist of a yellow background with black paragraph text and a title text in red. Then the resulting background layer contains the yellow color only. The foreground layer contains the black text color where the paragraph text is located and the red text color where the title is located. In the mask, pixels for which the foreground layer should be displayed are set to 1, the others are set to 0. I.e. the mask contains 1's where the black and the red text is and 0's everywhere else.

In the resulting TIFF the foreground layer, the background layer and the mask are stored as three images. The mask can be compressed differently to the foreground and background layer. Since all the detailed features have been moved to the mask, it makes sense to down-sample the foreground and background layers and use a low image quality. The mask on the other hand is usually stored with a lossless compression type optimized for text.

Note: Depending on the size of the input image it is possible that the algorithm decides that the whole input image consists of one photographic region covering the whole image. In this case, this second phase is omitted.

Note: The isolated pictures from phase 1 are not down-sampled nor compressed.

4.6.4 Phase 3: PDF Construction

In this phase the TIFF-file that resulted from phase 1 (recognition of photographic pictures and recognition of mask) and phase 2 (the segmentation into foreground and background layers and into a mask) is now used to construct a PDF-file by using `tiff2pdf`. If in phase 1, a single photographic region covering the entire image is detected, then the original image is used and the reconstruction is finished. Otherwise, the construction first places the background layer, followed by the foreground layer with the mask. Finally, if any isolated photographic pictures are found they are placed at their respective locations on top of the foreground layer.

4.7 Internal Engine

The internal engine comprises recognition functionality for the `tiffocr` tool, such as mask recognition or recognition of photographic picture regions.

In general, the string parameter for the OCR engines is composed by a sequence of Key-Value pairs that are separated by semicolons (;). In order to form the string parameter the following keys are supported by the internal engine.

4.7.1 RecognizeBlankPages

Key: `RecognizeBlankPages` Type: `Boolean` Default: `false`

Recognize blank pages of a certain file. A blank page is considered to be a page with a uniform coloring containing only slight noise. Colored, grayscale and bi-tonal pages can be subject to blank page recognition. The value of the Key-Value pair takes either `True` or `False`.

Example: Choose internal engine, recognize blank pages and store the recognition information (i.e. information about which pages are recognized as blank) in output.xml.

```
tiffocr -ocr "internal" -ocp "RecognizeBlankPages=true" -ocx "output.xml"
```

Blank pages can then be removed from a tiff file using `tiffmerge -rp`.

4.7.2 BlankPageMargin

Key: `BlankPageMargin` Type: `double` Default: `0.02`

Set the ratio the margin takes with respect to the corresponding page length. The margin is excluded from the analysis whether a page is blank. The allowed values range from 0 to 0.5. This parameter is only active if at the same time the value of `RecognizeBlankPages` is `True`.

Example: Choose internal engine and analyze if a page is blank without taking into account a margin of 5% on every side.

```
tiffocr -ocr "internal" -ocp "RecognizeBlankPages=true;BlankPageMargin=0.05"
```

Blank pages can then be removed from a tiff file using `tiffmerge -rp`.

4.7.3 DisableMaskEmbedding

Key: `DisableMaskEmbedding` Type: `Boolean` Default: `false`

If this option is set to `True`, no mask is embedded in the output TIFF. If this option is not set, a mask is embedded by default. The value of the Key-Value pair takes either `True` or `False`.

Example: Choose internal engine, recognize photographic pictures, but no mask shall be embedded in out.tif.

```
tiffocr -ocr "internal" -ocp ^
```

```
  "DisableMaskEmbedding=true;RecognizePictures=true" in.tif out.tif
```

4.7.4 RecognizePictures

Key: RecognizePictures Type: Boolean Default: false

Recognize photographic picture regions.

Example: Choose internal engine and recognize photographic pictures. The information about the recognized regions is then stored in output.xml.

```
  tiffocr -ocr "internal" -ocp "RecognizePictures=true" -ocx
"output.xml"
```

The recognition of photographic picture regions works only for colored pictures.

5 Interface Reference

5.1 General Information for all Tools

The following switches are present in all shell tools and have the same function.

5.1.1 `-lk` Set License Key

Set License Key `-lk ‹key›`

Pass a license key to the application at runtime instead of using one that is installed on the system.

```
tiff2pdf -lk X-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX ...
```

This is required in an OEM scenario only.

5.1.2 `-v` Verbose Mode

Verbose Mode `-v`

This option turns on the verbose mode.

In the verbose mode, additional information during the processing is written to the shell.

5.1.3 Return Codes

All return codes other than `0` indicate an error in the processing.

Return Codes

Value	Description
`0`	Success.
`1`	Couldn't open input file.
`3`	Error with given options, e.g. too many parameters.
`4`	Error while processing the files.
`10`	License error, e.g. invalid license key.
`13`	Unknown exception (e.g. memory exhausted).

5.1.4 Compression Types

The following compression types can be set.

Value	Compression	Lossy
1	Uncompressed	No
2	CCITT Group3	No
3	CCITT Group3 2D	No
4	CCITT Group 4	No
5	LZW	No
6	JPEG	Yes
7	JPEG Tech. Note #2	Yes
8	Adobe Deflate	No
32773	PackBits	No
32946	Deflate	No
34712	JPEG2000	Yes
34715	JBIG2	No[1]

5.2 TIFF to PDF/A-2 Conversion

The `tiff2pdf` tool converts a number of multi-page TIFF files into a PDF/A-2 conforming document.

If a TIFF page contains mixed raster content (MRC) layers (background, foreground, mask) they are converted into equivalent objects in the PDF document. The same is true for TIFF pages which contain alpha channels.

If a TIFF page contains embedded OCR text it is converted into an invisible text layer to make the resulting PDF document searchable. Note that only embedded OCR text is supported that was created using the `tiff2pdf` tool. Tools which create embedded OCR text from other vendors are not supported, since the embedding of OCR is not part of the TIFF V6 standard and all vendors are using proprietary tags.

If a TIFF page contains XMP metadata stream it is added as a Metadata entry in the PDF page object.

In addition to the device specific color spaces the `tiff2pdf` tool also supports calibrated color spaces and ICC color profiles.

The `tiff2pdf` tool minimizes the re-compression of streams. If the page is not compressed it will not be compressed in the resulting output. However, if the page uses LZW compression it will be re-compressed using FLATE, since LZW is not allowed in PDF/A-2.

Usage `tiff2pdf ‹option› in1.tif in2.tif ... out.pdf`

The tool can process any number of input files (all except the last parameter) and convert them to a single output file (the last parameter). The input parameters may contain wildcards.

[1] Lossy JBIG2 compression is not supported.

5.2.1 -d Define default resolution

Define default resolution `-d ‹dpi›`

If a TIFF page does not contain resolution tags they are defined by the parameter value of this option. The value's unit is dots per inch (DPI).

Example: Define the default resolution as 96 DPI.

```
tiff2pdf -d 96 in.tif out.pdf
```

5.2.2 -u Define user unit

Define user unit `-u ‹num›`

A PDF page supports a maximum width and height of 14'400 units. The default unit is one point (1/72 inch). If the TIFF page has larger dimensions then it may be helpful to use larger user units. The parameter value defines the user unit in multiple of points.

Example: Define the user unit as 1 inch.

```
tiff2pdf -u 72 in.tif out.pdf
```

5.2.3 -ocg Create optional content groups

Create optional content groups `-ocg`

PDF/A-2 supports optional content sometimes referred to as layers. If this option is used in conjunction with MRC layers, then the output document contains the optional content groups "Colored Text", "Black Text", "Foreground" and "Background" which can be individually turned on and off.

Example: Create optional content groups.

```
tiff2pdf -ocg in.tif out.pdf
```

5.2.4 -cl Set Conformance Level

Set Conformance Level `-cl ‹level›`

Set the PDF conformance level. Supported conformance levels are:

- pdf1.x Regular PDF versions such as 1.4, 1.5, 1.6, 1.7
- pdf2.0 Regular PDF version 2.0
- pdfa-1b PDF/A-1b format
- pdfa-1a PDF/A-1a format (accessibility)
- pdfa-2b PDF/A-2b format

- pdfa-2u PDF/A-2u format (Unicode)
- pdfa-2a PDF/A-2a format (accessibility)
- pdfa-3b PDF/A-3b format
- pdfa-3u PDF/A-3u format (Unicode)
- pdfa-3a PDF/A-3a format (accessibility)

The default is pdfa-2a.

Example: To create a document that conforms to PDF/A-2b, use a setting like this:

```
tiff2pdf -cl pdfa-2b input.tif output.pdf
```

Selecting a PDF/A conformance level will automatically generate the XML metadata and other requirements to meet the PDF/A specification. If JPEG2000 images are to be converted to PDF/A and the JPEG2000 compression shall be retained, a PDF/A-2 or PDF/A-3 conformance level must be selected.

5.2.5 `-cs` Set Default Color Space

Set Default Color Space `-cs ‹profile›`

This color space is used in the output PDF for pages that have no ICC profile available in the corresponding page in the input TIFF. A default color space profile can be set for both RGB and CMYK.

Example: Convert TIFF document into PDF document and set default color space:

```
tiff2pdf -cs "C:\\Windows\\System32\\spool\\drivers\\color\\sRGB Color Space Profile.icm"
input.tif output.pdf
```

5.2.6 `-ax` Set XMP Metadata

Set XMP Metadata `-ax ‹file›`

Specify a file with XMP metadata, which are added to the output document. The XMP metadata are copied only and not checked for conformance.

Example: Add XMP metadata from `file.xmp` to output document:

```
tiff2pdf -ax file.xmp input.tif output.pdf
```

5.2.7 `-io` Ignore OCR data

Ignore OCR data `-io`

Ignore OCR data that was recognized by `tiffocr`. Photographic image regions are not ignored by this option.

Example: Convert TIFF document into PDF document without embedding existing OCR text:

```
tiff2pdf -io input.tif output.pdf
```

5.2.8 -oi Set Output Intent

Set Output Intent `-oi ‹profile›`

The output intent holds the output color profile. Color profiles are usually provided with the OS. On Windows for example they can be found at `C:\Windows\System32\spool\drivers\color`.

Alternatively profiles can be found here:

- www.pdf-tools.com/public/downloads/resources/colorprofiles.zip
- www.color.org/srgbprofiles.html
- www.adobe.com/support/downloads/iccprofiles/iccprofiles_win.html

> **Note:** Most color profiles are copyrighted, therefore you should read the license agreements on the above links before using the color profiles.

Example: Set the output intent to a specific profile that exists on the system.

```
tiff2pdf -oi "C:\Windows\system32\spool\drivers\color\sRGB Color Space
Profile.icm" input.tif output.pdf
```

5.3 TIFF Compression

The `tiffcompress` tool compresses the pages of the input file with the selected compression algorithms.

For each of the TIFF classes Binary, Grayscale, Lab, Palette, RGB & YCbCr, Separated (CMYK) and MRC a different compression algorithm can be specified. The tool also allows for specifying a class specific quality measure for lossy compression algorithms.

The `tiffcompress` tool allows for upgrading embedded JPEG streams in the TIFF V6 format to the newer format specified in the independent JPEG group's Technical Note #2.

`tiffcompress` provides the possibility of using the mixed raster content (MRC) technique, i.e. continuous images can be converted into background, foreground and mask layers (Section 4.6). The computation of a mask is done in the `tiffocr` tool. If no mask is present, `tiffcompress` computes one when performing MRC. Photographic picture regions that are recognized by `tiffocr` are not subject to MRC. The foreground and background layers can additionally be down-sampled.

Usage `tiffcompress ‹options› in.tif out.tif`

The tool can only process one input single- or multi-page TIFF file. The output file contains the same number of pages but with differently compressed image data. If MRC layers are being created, the output file contains for every page images for the mask, the foreground and background layer, and the isolated photographic pictures that were recognized in `tiffocr`, see TIFF OCR (Optical Character Recognition).

5.3.1 Compression algorithm

Binary compression `-cb ‹n›`

Grayscale compression `-cg ‹n›`

Lab compression `-cl ‹n›`

Paletted compression `-cp ‹n›`

RGB & YCbCr compression `-cr ‹n›`

Separated (CMYK) compression `-cs ‹n›`

MRC compression `-cm ‹n›`

This set of switches is used to specify a compression algorithm specifically for each TIFF class. The second character in the switch is an abbreviation of the class as follows:

Value	Class
b	Binary
g	Grayscale
l	Lab
p	Palette
r	RGB & YCbCr
s	Separated (CMYK)
m	MRC Background and Foreground Layer

For supported compression types and corresponding values for parameter `‹n›`, see Compression Types.

Example: Specify LZW as the compression algorithm for RGB images.

```
tiffcompress -cr 5 in.tif out.tif
```

5.3.2 Compression quality

Binary compression quality `-qb ‹n›`

Grayscale compression quality `-qg ‹n›`

Lab compression quality `-ql ‹n›`

RGB & YCbCr compression quality `-qr ‹n›`

Separated (CMYK) compression quality `-qs ‹n›`

MRC compression quality `-qm ‹n›`

These switches specify the compression quality for the corresponding TIFF class. The value range is from 1 to 100. The quality is effective only when using a lossy compression algorithm (see table) for the specific class. As with the compression algorithm the second character of the switch denotes the class.

Example: Specify the JPEG compression algorithm with a quality measure of 75 for RGB images.

```
tiffcompress -cr 6 -qr 75 in.tif out.tif
```

5.3.3 Downsampling factor

Binary downsampling factor `-db ‹n›`

Gray scale downsampling factor `-dg ‹n›`

Lab downsampling factor `-dl ‹n›`

RGB & YCbCr downsampling factor `-dr ‹n›`

Separated (CMYK) downsampling factor `-ds ‹n›`

MRC layer downsampling factor `-dm ‹n›`

These switches specify the down sampling factor for the corresponding TIFF class. A value of 2 e.g. means that the resulting image dimension and resolution are divided by two. The factor is effective only if the down sampling filter implements the specific TIFF image. As with the compression algorithm the second character of the switch denotes the class.

Example: Specify a down sampling factor of 2 for RGB images.

```
tiffcompress -dr 2 in.tif out.tif
```

5.3.4 `-r` Recompress JPEG, JPEG2000 and JBIG2 streams

Recompress JPEG, JPEG2000 and JBIG2 streams `-r`

This switch recompresses streams that have already been compressed with a lossy algorithm (JPEG, JPEG2000, JBIG2) in order to reduce the size by specifying either a different algorithm or a different quality.

Example: Recompress RGB images using a JPEG2000 algorithm and a quality of 80.

```
tiffcompress -r -cr 34712 -qr 80 in.tif out.tif
```

5.3.5 `-u` Upgrade embedded JPEG streams from V6 to TN#2

Upgrade embedded JPEG streams from V6 to TN#2 `-u`

This switch re-writes TIFF pages containing JPEG streams according to the TIFF V6 specification in the newer format conforming to the Technical Note #2 of the independent JPEG group. The re-formatting does not involve any re-compression and thus is lossless.

Example: Upgrade the embedding format of JPEGs to conform to the TN#2 specification.

```
tiffcompress -u in.tif out.tif
```

5.3.6 Create mixed raster content (MRC) layers

Perform mixed raster content (MRC) segmentation	`-s`
Binarization threshold	`-st ‹n›`

These switches create layers according to the Mixed Raster Content (MRC) capabilities of the TIFF specification.

The separation of the pixels on the page into foreground layer and background layer is done by using a threshold value. (Black is 0, White is 255). The information which layer a pixel is associated with is stored in the mask. The threshold value should be set near to the average gray level of the image to achieve the best compression quality and ratio. The default threshold value is computed automatically from the image sample data.

The option `-s` separates the image into background, foreground and mask layer.

The foreground and background layer can usually be down-sampled as they contain low resolution data such as the color of the text. By means of the `-dm` switch a down-sampling factor can be specified. Isolated photographic pictures remain untouched under the MRC separation step.

Example: Create MRC layers using a threshold of 250 and a down-sampling factor of 2.

```
tiffcompress -s -st 250 -dm 2 in.tif out.tif
```

5.4 TIFF OCR (Optical Character Recognition)

The `tiffocr` tool recognizes text using an OCR engine and embeds it. Furthermore a binary mask is calculated and photographic picture regions are detected.

In order to use text recognition one of the supported OCR engines must be used. Currently the engines FineReader 10 and 11 from ABBYY are supported. These engines are available as separate product kits.

Furthermore there is an internal engine which provides some recognition functionality (see Section 4.7), such as recognition of a binary mask or recognition of photographic picture regions.

Mask computation is done by default. An embedded mask can then be used in the `tiffcompress` tool to perform MRC (Section 4.6).

The tool processes an input single- or multi-page TIFF file. The output file contains the same number of pages. If the OCR process for a specific page succeeds then the original image is replaced by the one delivered by the OCR engine, if any. The new image is compressed using a lossless algorithm such as CCITT G4 and LZW depending on the image class.

5.4.1 `-le` List available OCR engines

List available OCR engines	`-le`

This switch lists the set of available OCR engines. An engine can be selected out of this set to perform the text recognition.

5.4.2 Select OCR engine

OCR engine name `-ocr ‹engine›`

OCR engine parameters `-ocp ‹params›`

OCR engine languages `-ocl ‹langs›`

OCR result file name `-ocx ‹name›`

This set of switches allows for selecting an OCR engine, passing parameters to it and specifying an optional XML output file.

With the `-ocr` switch one of the supported OCR engines can be selected. Currently the values `"abbyy10"`, `"abbyy11"` and `"service"` are supported. In order to function correctly the tool requires the selected engine add-in to be installed.

In addition there is the internal engine `"internal"` available. To use this engine an add-in is not required. For more information see Section 4.7.

The `-ocp` switch is an engine specific string parameter. For more information on this, refer to the documentation of the engine add-in.

The `-ocl` switch receives a language identifier. The language is a hint for the OCR engine to recognize the text in the given language.

The `-ocx` switch creates an XML file containing the recognized text and the photographic picture regions. The XML format is documented in Appendix A.

5.4.3 Perform binarization prior to the text recognition

Perform binarization prior to OCR `-sb`

Binarization threshold `-st ‹n›`

The `-sb` switch converts color and gray scale image data to bi-tonal samples before sending the data to the OCR engine. The `-st` switch defines the black/white threshold (default: 128).

5.5 TIFF Image Import

The `tiffimp` tool imports JPEG, TIFF or PDF images and embeds them into a TIFF container.

The JPEG streams are embedded conforming to the TIFF V6 specification. The tool supports JPEG streams conforming to the JFIF and Adobe Photoshop format by interpreting the corresponding application specific markers.

If the input stream contains ICC profiles or XMP metadata streams they are embedded separately using the appropriate tags. All other application specific markers are ignored.

If the resolution information is missing in the JPEG stream a default resolution is used. The color space is retrieved from the application specific markers and, if missing, from the number of color channels.

Usage `tiffimp ‹options› in1.jpg in2.jpg ... out.tif`

The tool can process any number of input files (all except the last parameter) and convert them to a single output file (the last parameter). The input parameters may contain wildcards.

5.5.1 -d Define default resolution

Define default resolution `-d ‹dpi›`

If a TIFF page does not contain resolution tags they are defined by the parameter value of this switch. The value's unit is dots per inch (DPI).

Example: Define the default resolution as 96 DPI.

```
tiffimp -d 96 in.jpg out.jpg
```

5.5.2 -u Upgrade embedded JPEG streams from V6 to TN#2

Upgrade embedded JPEG streams from V6 to TN#2 `-u`

This switch re-writes TIFF pages containing JPEG streams according to the TIFF V6 specification in the newer format conforming to the Technical Note #2 of the independent JPEG group. The re-formatting does not involve any re-compression and thus is lossless.

Example: Upgrade the embedding format of JPEGs to conform to the TN#2 specification.

```
tiffimp -u in.tif out.tif
```

5.6 TIFF Merge

The `tiffmerge` tool merges single- and multi-page TIFF files into a large multi-page file without further processing except for embedded JPEG streams.

There is a special processing of embedded V6 JPEG streams. Instead of just copying the corresponding tags to the output the JPEG stream is re-assembled from the corresponding tags and then embedded into the output page conforming to the TIFF V6 specification. This fixes known problems with badly created TIFF V6 files (see Technical Note #2 for more information on this) and increases the interoperability with the most common readers (viewers) in use.

Usage `tiffmerge ‹options› in1.tif in2.tif ... out.tif`

The tool can process any number of input files (all except the last parameter) and convert them to a single output file (the last parameter). The input parameters may contain wildcards.

5.6.1 -rp Remove blank pages

Remove blank pages `-rp`

If this option is set pages that have been recognized by `tiffocr` to be blank are removed from the TIFF file.

Example: Remove blank pages recognized by `tiffocr`.

```
tiffmerge -rp in.tif out.tif
```

5.7 TIFF Split

The `tiffsplit` tool splits multi-page TIFF files into a set of single-page files without further processing except for embedded JPEG streams.

There is a special processing of embedded V6 JPEG streams. Instead of just copying the corresponding tags to the output the JPEG stream is re-assembled from the corresponding tags and then embedded into the output page conforming to the TIFF V6 specification. This fixes known problems with badly created TIFF V6 files (see Technical Note #2 for more information on this) and increases the interoperability with the most common readers (viewers) in use.

Usage `tiffsplit ‹options› in.tif ‹format›.tif`

The tool can only process a single input file and produce as many output files as the input file contains pages.

The name of the output file can be created using a format string. The syntax of the format string is a sub-set of the syntax of the "sprintf" function of the C programming language (see corresponding documentation for more information). Here are some examples:

Example: Creates the file names out_1.tif, out_2.tif, etc.

```
tiffsplit in.tif out_%d.tif
```

Example: Creates the file names with 3 digits and leading zeros.

```
tiffsplit in.tif out%03d.tif
```

Example: Creates the file names with 4 hexadecimal digits.

```
tiffsplit in.tif out_%04X.tif
```

5.8 TIFF Extract

The `tiffextract` tool allows to extract ICC color profiles and XMP metadata from a multi-page TIFF file. OCR data can be extracted from TIFF files that beforehand had been subject to OCR in `tiffocr`.

Usage `tiffextract ‹options› input1.tif input2.tif, ...`

The tool can process any number of input files. Furthermore, the input parameters may contain wildcards. (see Section 4.3)

5.8.1 `-ocv` Version of OCR XML format

Version of OCR XML format `-ocv ‹version›`

Set version of the format of the extracted OCR XML. Use this option together with `-xo`. By default, the stored value from the TIFF is used. In that case, no reformatting is performed. Current available versions are 1, 2 and 3.

Example: Perform OCR on input.tif and extract OCR data in format version 3.

```
tiffocr -ocr abbyy11 input.tif outputOcr.tif
tiffextract -xo -ocv 3 outputOcr.tif
```

5.8.2 Extraction

Extract ICC color profile `-xc`

Extract OCR data `-xo`

Extract metadata in XMP format `-xx`

Extract ICC color profiles, OCR data or metadata in XMP format. The generated output files are called: `<file-name>_<pagenumber>.<type>`, where `<type>` can either be `icc`, `xmp` or `ocr`.

Example: Extract OCR data in format version 2 from input1.tif and input2.tif.

```
tiffextract -xo -ocv 2 input1.tif input2.tif
```

Example: Extract ICC color profile from every page of input.tif.

```
tiffextract -xc input.tif
```

Example: Extract XMP metadata from all .tif files in directory with filename starting with input.

```
tiffextract -xx input*.tif
```

5.9 TIFF Scan

The `tiffscan` tool scans images from a TWAIN or WIA scanner into one multi-page file.

Custom scan settings can be specified in a user interface and can be saved in a capability file.

Usage `tiffscan ‹options› out.tif`

5.9.1 `-cl` load capability file

load capability file `-cl ‹file›`

Set capability file which specifies scan settings.

5.9.2 `-cs` save capability file

save capability file `-cs ‹file›`

Save scan configuration in capability file.

Example: Specify scan settings in a user interface and save them in `caps.dat`.

```
tiffscan -u -cs caps.dat
```

5.9.3 `-l` list sources

list sources `-l`

List all available scanners. Only scanners that match the architecture of `tiffscan.exe` are listed.

5.9.4 `-s` scanner product name

scanner product name `-s ‹name›`

Set specific TWAIN or WIA scanner. A list of available scanners can be obtained with switch `-l`.

Example: Set capability file `caps.dat` and choose "TWAIN2 FreeImage Software Scanner" for scanning. The output is stored in `outScan.tif`.

```
tiffscan -s "TWAIN2 FreeImage Software Scanner" -cl caps.dat outScan.tif
```

5.9.5 `-u` show user interface

show user interface `-u`

Show user interface that allows to configure scan settings.

Example: Configure scan settings in user interface and scan to `outScan.tif`.

```
tiffscan -u outScan.tif
```

6 Version History

6.1 Changes in Version 6

Shell `tiffextract`

- **Improved** OCR XML format. Maximum and default version increased to 4.

6.2 Changes in Version 5

- **New** additional supported operating system: Windows Server 2019.
- **Changed** behavior when reading a TIFF. The value `Relative` from tag `ResolutionUnit` is now interpreted as `Inch`.

Shell `tiffcompress`

- **Improved** MRC background and foreground layer coloring.

6.3 Changes in Version 4.12

- **New** HTTP proxy setting in the GUI license manager.
- **New** shell tool `tiffscan` to scan images from a TWAIN or WIA scanner.

Shell `tiffocr`

- **New** key `BlankPageMargin` for the formation of the OCR parameter used in `-ocp`. The corresponding value denotes the relative margin. The margin is excluded from the analysis if a page is blank.

Shell `tiffextract`

- **New** option `-ocv` to produce a fixed version of the OCR XML format.
- **New** included description of `tiffextract` in `TiffToolboxShell.pdf`.

Shell `tiffscan`

- **New** option `-cl` to load capability file.
- **New** option `-cs` to save capability file.
- **New** option `-l` to list sources.
- **New** option `-s` to set a scanner.
- **New** option `-u` to show user interface in order to configure scan settings.

6.4 Changes in Version 4.11

- **New** support for reading and writing PDF 2.0 documents.

- **Changed** OCR XML format (version 3)

6.5 Changes in Version 4.10

- **Changed** Product name from "3-Heights™ TIFF Tool Suite" to "3-Heights™ TIFF Toolbox Shell".

6.6 Changes in Version 4.9

- **Improved** metadata generation for standard PDF properties.

Shell `tiff2pdf`

- **New** option `-cs`: Set default ICC profile for device-specific color spaces. Switch can be set for one RGB and one CMYK color space.

6.7 Changes in Version 4.8

Shell `tiff2pdf`

- **New** option `-oi` to set output intent.

7 Licensing, Copyright, and Contact

PDF Tools AG is a world leader in PDF (Portable Document Format) software, delivering reliable PDF products to international customers in all market segments.

PDF Tools AG provides server-based software products designed specifically for developers, integrators, consultants, customizing specialists and IT-departments. Thousands of companies worldwide use our products directly and hundreds of thousands of users benefit from the technology indirectly via a global network of OEM partners. The tools can be easily embedded into application programs and are available for a multitude of operating system platforms.

Contact

PDF Tools AG
Brown-Boveri-Strasse 5
8050 Zürich
Switzerland
http://www.pdf-tools.com
pdfsales@pdf-tools.com

A OCR XML Format

A.1 Versions

The XML format has evolved over time and will continue to do so. Incompatible changes are denoted by increasing the format version. The current version is 4.

If applicable, the minimum format version of attributes or child elements is specified in parentheses.

The addition of new optional attributes is not considered an incompatible change. Applications that consume the XML must therefore be prepared to ignore unknown attributes.

A.2 Elements

A.2.1 <document> Element

The root element of the XML.

This element is omitted if the XML is describing a single page only. In that case, the <page> element is the root element.

Attributes:

`version` (**optional** in v1, **required** otherwise) The version of the XML format.

Default value is `"1"`.

Child elements:

<page> (**1..n**)

A.2.2 <page> Element

A single page that represents the recognized image.

Attributes:

`version` (**optional**) The version of the XML format.

If no version is specified, the value is inherited from the parent <document> element, if present. Default value is `"1"`.

`bb` The bounding box of the page in pixels: `"0 0 w h"`

`res` The resolution of the image.

Child elements:

<page-content> (**1**)

A.2.3 <page-content> Element

The root element of the page content.

Attributes:

tf (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v1:
<image> (**0..n**), <text> (**0..n**),

Child elements v2:
<image> (**0..n**), <barcode> (**0..n**) <text> (**0..n**),

Child elements v3:
 (**0..n**), <image> (**0..n**), <barcode> (**0..n**) <table> (**0..n**), <heading> (**0..n**), <paragraph> (**0..n**), <text> (**0..n**),

Child elements v4:
<header> (**1**), <footer> (**1**), <section> (**0..n**), <incut-group> (**0..n**), <incut> (**0..n**), <footnote> (**0..n**), <image> (**0..n**), <barcode> (**0..n**) <table> (**0..n**), <text-block> (**0..n**),

A.2.4 <header> Element

The page header.

Attributes:
tf (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v4:
<image> (**0..n**), <barcode> (**0..n**) <table> (**0..n**), <text-block> (**0..n**),

A.2.5 <footer> Element

The page footer.

Attributes:
tf (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v4:
<image> (**0..n**), <barcode> (**0..n**) <table> (**0..n**), <text-block> (**0..n**),

A.2.6 <section> Element

A page section.

Attributes:
tf (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v4:
<incut-group> (**0..n**), <incut> (**0..n**), <footnote> (**0..n**), <image> (**0..n**), <barcode> (**0..n**) <table> (**0..n**), <text-block> (**0..n**),

A.2.7 <artifact> Element

Content elements that cannot be classified.

Attributes:

tf (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v4:

<image> (**0..n**), <barcode> (**0..n**) <table> (**0..n**), <text-block> (**0..n**),

A.2.8 <incut-group> Element

A group of incuts.

Attributes:

tf (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v4:

<incut> (**0..n**),

A.2.9 <incut> Element

An incut.

Attributes:

tf (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v4:

<image> (**0..n**), <barcode> (**0..n**) <table> (**0..n**), <text-block> (**0..n**),

A.2.10 <footnote> Element

A footnote.

Attributes:

tf (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v4:

<image> (**0..n**), <barcode> (**0..n**) <table> (**0..n**), <text-block> (**0..n**),

A.2.11 Element

A generic group of content elements.

Attributes:

tf (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v3:

<text> (**0..n**), (**0..n**), <heading> (**0..n**), <paragraph> (**0..n**), <table> (**0..n**), <image> (**0..n**), <barcode> (**0..n**)

A.2.12 <caption> Element

The caption of an image or table.

Attributes:
tf (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v4:
<image> (**0..n**), <barcode> (**0..n**) <table> (**0..n**), <text-block> (**0..n**),

A.2.13 <text-block> Element

A block of text.

Attributes:
tf (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v4:
<list> (**0..n**), <heading> (**0..n**), <paragraph> (**0..n**), <word> (**0..n**), <text> (**0..n**),

A.2.14 <list> Element

A list.

Attributes:
tf (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v4:
<item> (**0..n**),

A.2.15 <item> Element

An item in a list.

Attributes:
tf (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v4:
<list> (**0..n**), <heading> (**0..n**), <paragraph> (**0..n**), <word> (**0..n**), <text> (**0..n**),

A.2.16 <heading> Element

A text heading.

Attributes:
tf (**optional**), bb (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v1-v3:
<text> (**0..n**)

Child elements v4:
<word> (**0..n**) <text> (**0..n**),

A.2.17 <paragraph> Element

A text paragraph.

Attributes:
tf (**optional**), bb (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v1-v3:
<text> (**0..n**)

Child elements v4:
<word> (**0..n**) <text> (**0..n**),

A.2.18 <word> Element

A single word.

Attributes:
tf (**optional**), bb (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v4:
<text> (**0..n**),

A.2.19 <text> Element

A text fragment.

The base line of the text is determined by the line $y = 0$ in the transformed coordinate system. Usually, the transformation looks like `tf="1 0 0 1 x y"`, where (x, y) is the baseline position of the first character.

The baseline and the bounding box are not necessarily intersecting.

> **Note:** In version 1 of the format, barcodes are represented by <text> elements with `font-name="Barcode"` or `font-name="BarcodeHex"`.

Attributes:
tf (**required**), bb (**required**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**),

`suspicious-chars` (**optional**) The zero based indexes of all suspicious characters, separated by a space. If the attribute is empty, there are no suspicious characters.

If the attribute is missing, the information is unknown.

`char-left-pos` (**optional**) The position of the left border of each character, separated by a space. The position is measured on the baseline starting from the left border.

If the attribute is missing, the information is unknown.

`char-right-pos` (**optional**) The position of the right border of each character, separated by a space. The position is measured on the baseline starting from the left border.

If the attribute is missing, the information is unknown.

Text content:
The text content of the text fragment as plain text.

A.2.20 <table> Element

A table.

Attributes: tf (**optional**), font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements v1-v3:
<row> (**0..n**)

Child elements v4:
<row> (**0..n**) <caption> (**0..n**)

A.2.21 <row> Element

A table row.

Attributes: font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

Child elements:
<cell> (**0..n**)

A.2.22 <cell> Element

A table cell.

Attributes:
font-name (**optional**), font-family (**optional**), font-styles (**optional**), font-size (**optional**), locale (**optional**)

`type` (**optional**) The cell type.

`"data"` (Default) A data cell.

`"heading"` A header cell.

Child elements v3:
<text> (**0..n**), (**0..n**), <heading> (**0..n**), <paragraph> (**0..n**), <table> (**0..n**), <image> (**0..n**), <barcode> (**0..n**)

Child elements v4:
<image> (**0..n**), <barcode> (**0..n**) <table> (**0..n**), <text-block> (**0..n**),

A.2.23 <image> Element

An image.

Attributes:

tf (**optional**), bb (**optional**),

`type` (**optional**) The image type.

`"raster"` (Default) A raster picture.

`"vector"` A vector graphic.

Child elements v4:

<caption> (**0..n**)

A.2.24 <barcode> Element

An 1D or 2D barcode.

> **Note:** In version 1 of the format, barcodes are represented by <text> elements with `font-name="Barcode"` or `font-name="BarcodeHex"`.

Attributes:

tf (**optional**), bb (**optional**),

`encoding` (**optional**) The encoding of the barcode value.

`"hex"` Encoded as a hexadecimal string.

Text content:

The barcode value as plain text or in the specified encoding.

A.3 Common Attributes

A.3.1 tf Attribute

The coordinate transformation matrix of the element.

`tf="m11 m12 m21 m22 dx dy"`

The transformation matrix is specified by six numbers. All information about orientation, rotation, scaling, skewing and translation can be calculated based on these six numbers.

The actual matrix is $M = \begin{bmatrix} m11 & m12 & 0 \\ m21 & m22 & 0 \\ dx & dy & 1 \end{bmatrix}$

This matrix is used to define a transformation of a vector $[\,x\ y\ 0\,]$ to a vector $[\,x'\ y'\ 0\,] = [\,x\ y\ 0\,] \cdot M$, where (x, y) is the original point and (x', y') is the transformed point on the page.

A.3.2 bb Attribute

The bounding box of the element in the transformed coordinate system.

`bb="x y w h"`

If the element also contains the tf attribute, the transformation is applied, before the bounding box is computed.

A.3.3 font-name Attribute

The name of the font used for <text> elements.

The attribute is inheritable, i.e. it can occur on any parent element of the text, down to the <page-content>.

A.3.4 font-family Attribute

The family of the font used for <text> elements, separated by a space.

Possible values are:

"mono" A monospaced font, i.e. every character has the same width. An example of such a font is "Courier".

"sans" A font without serifs (sans serif). An example of such a font is "Arial".

"serif" A font with serifs. An example of such a font is "Times".

The attribute is inheritable, i.e. it can occur on any parent element of the text, down to the <page-content>.

A.3.5 font-styles Attribute

The list of styles of the font used for <text> elements, separated by a space.

Possible values are:

"bold"

"italic"

"underline"

"strikeout"

The attribute is inheritable, i.e. it can occur on any parent element of the text, down to the <page-content>.

A.3.6 font-size Attribute

The size the font used for <text> elements.

The attribute is inheritable, i.e. it can occur on any parent element of the text, down to the <page-content>.

A.3.7 locale Attribute

The locale of <text> elements in ISO format.

The attribute is inheritable, i.e. it can occur on any parent element of the text, down to the <page-content>.

A.4 Example

```
<?xml version="1.0" encoding="utf-8"?>
<page xmlns="http://www.pdf-tools.com/ocr" version="3" bb="0 0 2481 3508" res="300 300">
  <page-content font-name="Times New Roman" font-family="serif" font-styles=""
               font-size="13" locale="en-US">
    <div font-styles="bold" font-size="18">
      <heading bb="297 314 1879 381">
        <text tf="1 0 0 1 297 366" bb="0 -51 146 0" suspicious-chars=""
              char-left-pos="0 48 84 118" char-right-pos="41 80 112 146">Face</text>
      </heading>
      <paragraph bb="296 623 2110 727" font-size="12" font-styles="">
        <text tf="1 0 0 1 430 658" bb="0 -33 122 11" suspicious-chars=""
             char-left-pos="0 16 55 78 103" char-right-pos="14 54 74 100 122">Image</text>
        <text tf="1 0 0 1 568 658" bb="0 -35 87 11" suspicious-chars=""
              char-left-pos="0 25 45 71" char-right-pos="23 44 70 87">days</text>
      </paragraph>
    </div>
    <table font-name="Arial" font-family="sans" font-size="10">
      <row>
        <cell>
          <paragraph bb="299 1042 458 1071">
            <text tf="1 0 0 1 299 1071" bb="0 -29 76 0" suspicious-chars="1 2"
                  char-left-pos="0 20 32 46 63"
                  char-right-pos="15 24 43 60 76">First</text>
          </paragraph>
        </cell>
        <cell>
          <paragraph bb="935 1040 1138 1071">
            <text tf="1 0 0 1 935 1071" bb="0 -29 78 0" suspicious-chars=""
                  char-left-pos="0 25 58" char-right-pos="22 54 78">Two</text>
          </paragraph>
        </cell>
      </row>
      <row>
        <cell>
          <paragraph bb="297 1098 513 1129">
            <text tf="1 0 0 1 297 1129" bb="0 -31 131 0" suspicious-chars=""
                  char-left-pos="0 21 44 63 89 112"
                  char-right-pos="17 40 60 83 107 131">Second</text>
          </paragraph>
        </cell>
        <cell>
          <paragraph bb="937 1098 1217 1129">
            <text tf="1 0 0 1 937 1129" bb="0 -31 131 0" suspicious-chars=""
                  char-left-pos="0 21 44 64 89 112"
                  char-right-pos="17 40 60 84 107 131">Second</text>
          </paragraph>
        </cell>
      </row>
    </table>
    <image bb="293 1305 651 1767" type="raster"/>
    <barcode bb="514 1982 858 2057">0123456789</barcode>
  </page-content>
</page>
```