



PDFTron PDF/A Manager™ User Manual

Version 1.x

LEGAL STATEMENT AND COPYRIGHT NOTICE

PDFTron PDF/A Manager™ User Manual
 Part number: PDFTRON-1-PDFAManagerCMD
 Part number: PDFTRON-1-PDFAManagerSDK
 Last Updated: February 1, 2010

© 2009-2010 PDFTron Systems, Inc. All Rights Reserved.

All information contained herein is the property of PDFTron Systems, Inc. ("PDFTron"). No part of this publication (whether in hardcopy or electronic form) may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of PDFTron Systems, Inc..

The information in this publication is provided for informational use only, is subject to change without notice, and should not be construed as a commitment by PDFTron. PDFTron assumes no responsibility or liability for any loss or damage that may arise from the use of any information in this publication. The software described in this user manual is furnished under License (enclosed in the software package) and may only be used or copied in accordance with the terms of that License.

PDFTron and the names of PDFTron products referenced herein are either trademarks and/or service marks and/or registered trademarks of PDFTron Systems, Inc. PDFTron, PDFNet SDK, PDF/A Manager, PDF2Image, PDF2SVG, PDF2Text, PDF2XPS, XPSConvert, PDFTron PDFSecure, PDF PageMaster, CosEdit, PDFNet SDK, PDF2Image SDK, PDF2SVG SDK, PDF2Text SDK, PDF2XPS SDK, XPSConvert SDK, PDFSecure SDK, PDF PageMaster SDK and associated Logos are either trademarks and/or service marks and/or registered trademarks of PDFTron Systems, Inc.

Any other brand or product names mentioned in this publication are the registered trademarks or trademarks of their respective holders. Mention of a product in this document does not necessarily imply endorsement of the product.

3

5.2	Reporting Progress Messages and Errors	26
5.3	PDF/A Add-on module for PDFNet SDK	27
6.	Support	29
6.1	Reporting Problems	29
6.2	Contact Information	29

1. Introduction

1.1 An Introduction to PDFTron PDF/A Manager

PDF/A Manager is a PDF/A (ISO 19005-1) validation and conversion software. It is available as a command-line tool and as a development toolkit.

The conversion option analyses the content of existing PDF files and performs a sequence of modifications in order to produce a PDF/A compliant document. Features that are not suitable for long-term archiving (such as encryption, obsolete compression schemes, missing fonts, or device-dependent color) are replaced with their PDF/A compliant equivalents. Because the conversion process applies only necessary changes to the source file, the information loss is minimal. Also, because the converter provides a detailed report for each change, it is simple to inspect changes and to determine whether the conversion loss is acceptable.

The validation option in PDF/A Manager can be used to quickly determine whether a PDF file fully complies with the PDF/A specification according to the international standard ISO 19005-1. For files that are not compliant, the validation option can be used to produce a detailed report of compliance violations as well as a list of relevant error objects.

Like other PDFTron products, PDF/A Manager does not rely on other third-party software. PDF/A Manager can be used in server environments or as a batch conversion process. PDF/A Manager Command-Line utility is based on PDFNet SDK core technology which is available for integration in third party solutions and applications. For more information about PDFNet SDK, please contact a PDFTron representative or visit <http://www.pdftron.com/pdfnet> for more information.

1.1.1 Key Functions

- Checks if a PDF file is compliant with PDF/A (ISO 19005-1) specification.
- Converts any PDF to a PDF/A compliant document.
- Supports both PDF/A-1a and PDF/A-1b.
- Produces a detailed report of compliance violations and associated PDF objects.
- Keeps the required changes a minimum, preserving the consistency of the original.
- Tracks all changes to allow for automatic assessment of data loss.
- Allows user to customize compliance checks or omit specific changes during PDF/A conversion.
- Preserves tags, logical structure, and color information in existing PDF documents.
- Offers automatic font substitution, embedding, and subsetting options.
- Supports automation and batch operation. PDF/A Manager is designed to be used in unattended mode in high throughput server or batch environments.

1.1.2 Benefits - Why PDF/A Manager?

- PDF/A Manager is an indispensable tool for anyone working with PDF/A.
- Supports all versions of the PDF format (PDF 1.0 to ISO32000).
- Includes an option to create linearized (web-optimized) PDF/A documents.
- Fast, reliable and suitable for server use.

1.1.3 Common Use Case Scenarios

- Libraries, newspaper agencies, and government institutions can use PDF/A Manager to automate PDF archiving in high-throughput web applications of server based environments.
- Maintainers or large collections of existing PDF documents can use PDF/A Manager in batch mode to ensure that all files are suitable for long-term archiving.

- #### 1.1.4 Operating Systems Supported

- ### 1.1.5 System Requirements

- ## 1.2 About This Manual

- [Section 1](#) introduces PDF/A Manager and describes the manual.
- [Section 2](#) explains how to install and uninstall PDF/A Manager.
- [Section 3](#) summarizes the command-line arguments available for PDF/A Manager and covers basic usage.
- [Section 4](#) covers PDF/A Manager reporting.
- [Section 5](#) contains an overview and basic usage of PDF/A Manager SDK.
- [Section 6](#) is where you will find all the support information you may require, such as how to report a problem with the software.

C

ba
b
le
es
g
er

er,
SD

3. Overview

PDFTron's PDF/A Manager is a command-line application designed to validate and convert generic PDF documents to PDF/A (ISO 19005-1) compliant format. This section covers basic usage and the command string syntax, used both in the PDF/A Manager Command-Line Application as well as in PDF/A Manager SDK.

3.1 Basic Syntax

The basic command-line syntax is:

```
pdfa [options] file1 file2 folder1 file3 ...
```

3.2 Command-Line Summary

Option	Parameter	Description
-x or --noxml	--noxml	<p>Disable XML report generation.</p> <p>By default, PDF/A Manager will generate an XML report (report.xml) in the output folder with all of the details of the validation and conversion results. The XML file can be viewed within most modern web-browsers.</p> <p>If this parameter is specified, validation and conversion results will be reported as part of the console output.</p>
-o or --output	e.g. -o myfolder -o c:\foo -o ../../bar	<p>The output directory. The folder can be relative to the current working folder. If the folder does not exist, PDF/A Manager will attempt to create the required path. If this parameter is not specified, all files will be saved relative to the current working folder.</p>
--subfolders		<p>Process all sub-directory for every directory specified in the argument list. By default, sub-directories are not processed.</p>
-f or --fname		<p>The explicit name for the converted file. The parameter is used only when converting one file at a time. The output file will be stored in the output folder.</p>
-p or --pass	-p "my pass"	<p>The password to process the input file. The password is required only in the conversion mode and when the input document is encrypted.</p>
--suffix	--suffix "_PDF_A"	<p>A string to append to all output filenames. This option is typically used to tag output filenames during multi-file or batch conversion. For example, if input file is "my.pdf" and suffix is "_pdfa" the output filename will be "my_pdfa.pdf".</p>
-l or --level	-l B	<p>PDF/A Conformance Level: [A or B]</p> <p>Level A conforming files must adhere to all of the requirements of ISO 19005. A file meeting this conformance level is said to be a 'conforming PDF/A-1a file.'</p>

10

This section covers the basic usage of PDF/A Manager, explaining all the available options and illustrating examples.

By default, PDF/A Manager stores converted files in the current working folder. To specify another output location, use the '-o' (or --output) parameter. For example:

Note: If the specified path does not exist, PDF/A Manager will attempt to create the necessary folders.

By default, PDF/A Manager will create a single output PDF/A document based on every input PDF file. The output filename is constructed by appending a suffix to the name of the input PDF file. For example, 'my.pdf' will be saved as 'my_pdfa.pdf'. It is possible to change the suffix using --suffix option. For example:

3.3.3 How do I run PDF/A Manager in validation mode?

To run PDF/A Manager in validation mode simply omit '-c' or '--convert' from the command line. For example,

3.3.4 How do I convert/validate a password protected PDF?

PDF/A Manager will, without user intervention, decrypt and convert documents secured with a master/owner password. If the document is secured using a user (or 'file open') password, PDF/A Manager will prompt you to enter the password.

For unattended conversion, the password can also be specified directly on the command-line using the '-p' (or --password) option. For example:

The above command line will convert PDF to PDF/A and will use the provided password ('secret') to open the secured document (i.e. 'secured.pdf').

Note: PDF/A Manager supports all standard security options available in PDF, including 40 and 128 bit RC4 encryption, Crypt filters, and 128 AES (Advanced Encryption Standard) encryption.

PDF/A Manager supports batch conversion and validation of many PDF files in a single pass. To convert all PDF files in a given folder(s) you can use the following syntax:

The '--subfolders' option can be used to recursively process all subfolders. For example, the following line will convert all documents in 'myfolder1' and 'myfolder2' as well as all subfolders:

By default, PDF/A Manager will convert all files with the extension `'pdf'`. To select different files based on the extension use the `--extension` parameter. For example, to convert all PDF documents with a custom extension `'blob'`, you could use the following line:

Wildcard characters can also be used to process multiple input files.

```
C:\test1 >dir
Directory of C:\test1
01/04/2009    03:35 PM        <DIR>          .
01/04/2009    03:35 PM        <DIR>          ..
05/21/2009    02:27 PM                A1.pdf
05/03/2009    09:38 AM                A2.pdf
05/20/2009    08:46 AM                B1.pdf
05/15/2009    12:50 PM                B2.pdf
```

```
c:\>pdftk --noxml c:/test1/*.pdf
```

```
pdfta --noxml c:/test1/A*.pdf
```

```
pdftk --noxml c:/test1/*1.pdf
```

The wildcards are expanded in the same manner as operating system commands. (Please refer to your operating system user's guide if you are unfamiliar with wildcards). Enclosing an argument in double quotation marks (" ") suppresses the wildcard expansion. Within quoted arguments, you can represent quotation marks literally by preceding the double-quotation-mark character with a backslash (\). If no matches are found for the wildcard argument, the argument is passed literally.

3.4 Exit Codes

To provide additional feedback, PDF/A Manager returns exit codes after completing processing. The exit codes can be used to provide user feedback for logging, etc. This is particularly important for applications running in an unattended environment.

The following table lists possible exit codes and their description:

Exit Code	Description
0	All files converted successfully.
1	Document is secured. Need a valid password to open the document.
2	Error opening the input file(s).
3	An unknown exception encountered.

All codes other than '0' indicate that there was an error during the conversion process.

The following illustrates a sample Windows batch script that processes exit codes:

```
@echo off rem convert all PDF files in 'data' folder
pdfa ./data
if errorlevel 1 goto passwd
if errorlevel 2 goto inputerr
if errorlevel 3 goto othererror
if errorlevel 0 goto exit

:passwd
echo Document is protected. Need a valid password to open the document.
goto exit

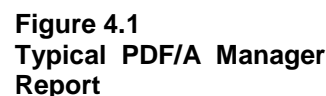
:inputerr
echo No input files specified.
goto exit

:othererror
echo An error encountered during processing.
goto exit

:exit
```

This section covers PDF/A Manager report formats and all the possible error conversion and validation options.

By default PDF/A Manager stores details of validation and conversion processing in an XML file. The report is saved as 'report.xml' in the output directory. Similarly, the style sheet for the report is saved as 'report.xsl' in the output directory. Double clicking on 'report.xml' will launch a web browser display HTML report as shown in the following figure:



The report includes the following information:

- The date and time when the report was generated
- A listing of files that fail validation prior to conversion. The information in this table includes the input PDF filename, error code, error message, and a list of relevant PDF object numbers.
- A listing of files that pass validation and for which no conversion is necessary.
- A listing of files that fail validation after to conversion. Usually this list includes files that can't be fully converted to PDF/A without significant information loss.
- A final list of files that pass validation after conversion.

The general structure of XML report is as follows:

```

<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="report.xsl"?>
<PDFAManagerReport RunDateTime="2009-Jul-16 22:46:04">
  <Validation>
    <Fail FileName="my.pdf" FileNameAndPath="c:\my.pdf">
      <Error Code="e_PDFA232" Message="Not a valid ICC color profile"
        Refs="3"/>
    </Fail>
  </Validation>
  <Validation>
    <Fail FileName="license.pdf" FileNameAndPath="c:\ license.pdf">
      <Error Code="e_PDFA185" Message="'obj' not followed by EOL marker"
        Refs="1, 2, 3, 4, 5, 6, 7, 8, 9, 12"/>
      <Error Code="e_PDFA341" Message="The font is not embedded"
        Refs="16, 17, 18"/>
      <Error Code="e_PDFA353" Message="Embedded TrueType font program..."
        Refs="16, 17, 18"/>
      <Error Code="e_PDFA723" Message="The XMP Metadata stream is not
        valid" Refs="39"/>
      <Error Code="e_PDFA2331" Message="Device-specific color space used,
        but no GTS_PDFA1 OutputIntent" Refs="1, 4, 14"/>
    </Fail>
  </Validation>
</PDFAManagerReport>

```

To disable report generation, simply include '--noxml' in the command-line string. Disabling XML report generation will direct processing results to the console window.

```
c:\PDFA\>pdfa --noxml -c -o OUT my.pdf
```

Processing...

VLD-[FAIL]: my.pdf

- e_PDFA185: 'obj' not followed by EOL marker
Obj Refs:1, 2, 3, 4, 5, 6, 7, 8, 9, 12
- e_PDFA341: The font is not embedded
Obj Refs:16, 17, 18
- e_PDFA723: The XMP Metadata stream is not valid
Obj Refs:39
- e_PDFA2331: Device-specific color space used, but no GTS_PDFA1 Output Intent
Obj Refs:1, 4, 14

CNV-[PASS]: my_pdfa.pdf

4.2 PDF/A Manager Validation and Conversion Error Codes

PDF/A Manager analyzes the content of existing PDF files and performs a sequence of modifications in order to produce a PDF/A compliant document. Features that are not suitable for long-term archiving (such as encryption, obsolete compression schemes, missing fonts, or device-dependent color) are replaced with their PDF/A compliant equivalents.

4.2.1 PDF/A Level B Validation Errors

The following table is a non-exhaustive list of checks and conversions performed by PDF/A Manager for Level B Validation:

Error Code	Description
e_PDFA0_1_1	Corrupt document
e_PDFA0_1_2	Corrupt content stream.
e_PDFA0_1_3	Using JPEG2000 compression (PDF 1.4 compatibility).
e_PDFA0_1_4	Contains compressed object streams (PDF 1.4 compatibility).
e_PDFA0_1_5	Contains cross-reference streams (PDF 1.4 compatibility).
e_PDFA1_2_1	Document does not start with % character.
e_PDFA1_2_2	File header line not followed by % and 4 characters > 127.
e_PDFA1_3_1	The trailer dictionary does not contain ID.
e_PDFA1_3_2	Trailer dictionary contains Encrypt.
e_PDFA1_3_3	Data after last EOF marker.
e_PDFA1_3_4	Linearized file: ID in 1st page and last trailer are different.
e_PDFA1_4_1	Subsection header: starting object number and range not separated by a single space.
e_PDFA1_4_2	'xref' and cross reference subsection header not separated by a single EOL marker.
e_PDFA1_6_1	Invalid hexadecimal strings used.
e_PDFA1_7_1	The 'stream' token is not followed by CR and LF or a single LF.
e_PDFA1_7_2	The 'endstream' token is not preceded by EOL.
e_PDFA1_7_3	The value of Length does not match the number of bytes.
e_PDFA1_7_4	A stream object dictionary contains the F, FFilter, or FDecodeParms keys.
e_PDFA1_8_1	Object number and generation number are not separated by a single

e_PDFA4_3	Transparency used (Form XObject with transparency group).
e_PDFA4_4	Transparency used (Blend mode is not 'Normal').
e_PDFA4_5	Transparency used ('CA' value is not 1.0).
e_PDFA4_6	Transparency used ('ca' value is not 1.0).
e_PDFA5_2_1	Unknown annotation type.
e_PDFA5_2_2	FileAttachment annotation is not permitted.
e_PDFA5_2_3	Sound annotation is not permitted.
e_PDFA5_2_4	Movie annotation is not permitted.
e_PDFA5_2_5	Redact annotation is not permitted.
e_PDFA5_2_6	3D annotation is not permitted.
e_PDFA5_2_7	Caret annotation is not permitted.
e_PDFA5_2_8	Watermark annotation is not permitted.
e_PDFA5_2_9	Polygon annotation is not permitted.
e_PDFA5_2_10	PolyLine annotation is not permitted.
e_PDFA5_2_11	Screen annotation is not permitted.
e_PDFA5_3_1	An annotation dictionary contains the CA key with a value other than 1.0.
e_PDFA5_3_2_1	An annotation dictionary is missing F key.
e_PDFA5_3_2_2	An annotation's 'Print' flag is not set
e_PDFA5_3_2_3	An annotation's 'Hidden' flag is set.
e_PDFA5_3_2_4	An annotation's 'Invisible' flag is set.
e_PDFA5_3_2_5	An annotation's 'NoView' flag is set.
e_PDFA5_3_3_1	An annotation's C entry present but no OutputIntent present
e_PDFA5_3_3_2	An annotation's C entry present but OutputIntent has non-RGB destination profile
e_PDFA5_3_3_3	An annotation's IC entry present but no OutputIntent present
e_PDFA5_3_3_4	An annotation's IC entry present and OutputIntent has non-RGB destination profile
e_PDFA5_3_4_1	An annotation AP dictionary has entries other than the N entry.

e_PDFA7_8_5	'pdfaExtension:property' not found.
e_PDFA7_8_6	'pdfaExtension:property' is using a wrong value type.
e_PDFA7_8_7	'pdfaProperty:name' not found.
e_PDFA7_8_8	'pdfaProperty:name' is using a wrong value type.
e_PDFA7_8_9	A description for a property is missing in 'pdfaSchema:property' sequence.
e_PDFA7_8_10	'pdfaProperty:valueType' not found.
e_PDFA7_8_11	The required namespace prefix for extension schema is 'pdfaExtension'.
e_PDFA7_8_12	The required field namespace prefix is 'pdfaSchema'.
e_PDFA7_8_13	The required field namespace prefix is 'pdfaProperty'.
e_PDFA7_8_14	The required field namespace prefix is 'pdfaType'.
e_PDFA7_8_15	The required field namespace prefix is 'pdfaField'.
e_PDFA7_8_16	'pdfaSchema:valueType' not found.
e_PDFA7_8_17	'pdfaSchema:valueType' is using a wrong value type.
e_PDFA7_8_18	Required property 'valueType' missing in PDF/A Schema Value Type.
e_PDFA7_8_19	'pdfaType:type' not found.
e_PDFA7_8_20	'pdfaType:type' is using a wrong value type.
e_PDFA7_8_21	'pdfaType:description' not found.
e_PDFA7_8_22	'pdfaType:namespaceURI' not found.
e_PDFA7_8_23	'pdfaType:field' is using a wrong value type.
e_PDFA7_8_24	'pdfaField:name' not found.
e_PDFA7_8_25	'pdfaField:name' is using a wrong value type.
e_PDFA7_8_26	'pdfaField:valueType' not found.
e_PDFA7_8_27	'pdfaField:valueType' is using a wrong type.
e_PDFA7_8_28	'pdfaField:description' not found.
e_PDFA7_8_29	'pdfaField:description' is using a wrong type.
e_PDFA7_8_30	Required description for 'pdfaField::valueType' is missing.
e_PDFA7_8_31	A property doesn't match its custom schema type.

5. Overview of PDF/A Manager SDK

Besides being offered as a command-line utility, PDF/A Manager is also available for integration with third party applications through two software development kits (SDKs): 'PDF/A Manager SDK' and 'PDFNet SDK PDF/A add-on module'.

PDF/A Manager SDK has a very simple API which resembles options in the PDF/A Manager Command-Line Application. The 'PDFNet SDK PDF/A add-on module' offers a more programmatic interface that is better integrated with other PDF processing options available in the PDFNet PDF library (<http://www.pdftron.com/pdfnet>).

5.1 Working with PDF/A Manager SDK

PDF/A Manager SDK offers a simple to use API for PDF/A validation and conversion. The SDK is available as a plain 'C DLL' and can be easily accessed from any programming language (including C#, VB.NET, C/C++, Java, VB6, Perl, Python, Ruby, Delphi, etc).

The entire API consists of only two functions: PDFAINit and PDFARun.

PDFAINit is called only once per process session to initialize the library and register the component. After initializing the library, PDFARun can be called many times to process PDF documents or folders with PDF documents.

The following is the simplest application that can be built using PDF/A Manager SDK:

```

// Using C# or C/C++
void main() {
    PDFAINit("username", "company", "lic_key");
    PDFARun("-o c:/out -c -z c:/test/tiger.pdf", null, new IntPtr(0));
}
  
```

This application essentially executes a hardcoded operation that converts 'tiger.pdf' to PDF/A Compliant format (i.e. tiger_pdfa.pdf stored in 'c:/out'). To convert all PDF documents in 'test' folder simply delete 'tiger.pdf' from the command string.

The first parameter of the PDFARun() function is a command string which is exactly the same as the general syntax used for the PDF/A Manager Command-Line application. For a detailed explanation of all options, please refer to section 'Basic Syntax' of this manual. The PDF/A Manager Command-Line application is a great tool to get to know all the available options. In fact, building a command-line application using PDF/A Manager SDK is as simple as the following listing:

```

// Using C#
static void Main(string[] args) {
    PDFAINit("username", "company", "lic_key");

    String s = "";
    foreach (string arg in args) {
        s += arg + " ";
    }
    PDFARun(s, null, new IntPtr(0));
}
  
```



```
// Using C#
static void Main(string[] args)
{
    PDFAMain(username, company, lic_key);

    // See the next section for more info regarding PDFACallback.
    PDFACallback mycallback = new PDFACallback(MyCallback);

    string output_folder = "Output";
    string open_password = "secret";

    bool noxml = false;        // Disable XML report generation?
    bool linearize = true;      // Linearize the converted files?
    bool convert = true;        // Convert or validate?
    bool revalidate = true;     // Revalidate after conversion?
    bool pdfa_1a = false;      // Level A or B?

    // -----
    // Given the above settings build a command string.
    string s = "";

    if (output_folder != "") s += "-o " + output_folder + " ";
    if (open_password != "") s += "-p " + open_password + " ";

    if (noxml) s += "--noxml ";
    if (linearize) s += "-z ";
    if (convert) s += "-c ";
    if (revalidate) s += "--nr ";
    if (pdfa_1a) s += "-l A ";

    // specify input PDF files and folders...
    s += "../.../.../license.pdf ";
    s += "Output ";
    // s += "c:/my_pdf_folder "; etc ...

    //an array list to store the saved file path(s)
    ArrayList saved_files=new ArrayList();

    //create a GCHandle to allow MyCallback to access
    //the object saved_files
    GCHandle gch = GCHandle.Alloc(saved_files);
    // Execute the command string.
    PDFARun(s, mycallback, (IntPtr)gch);
    gch.Free();

    //print the saved file(s)
    Console.WriteLine("The Saved Files Are:");
    for(int i=0; i<saved_files.Count; i++)
        Console.WriteLine(saved_files[i]);
}
```

To find out if PDF/A Manager processing was successful, the application can query the status code returned by `PDFARun()`.

```
int ret = PDFARun(...);
if (ret == PDFA_OK) {
    // No errors...
}
else if (ret == PDFERR_BADKEY ) {
    // bad license key...
}
else if (ret == PDF2A_ERR_DIRCREATE ) {
    // Failed to create the output directory
}
else {
    // Other error
}
```

For more detailed error and message reporting, you can pass a pointer to the custom callback function in the second parameter of `PDFARun()`. The last parameter in `PDFARun` is a pointer to custom data that you may want to pass to the callback function.

```
// Using C/C++
char* MyCallback(int mode, char* msg, void* user_data) {
    if (mode == PDFFA_ERROR) {
        cout << "Error: " << msg << endl;
    }
    else if (mode == PDFFA_MSG) {
        cout << msg;
    }
    else if (mode == PDFFA_GETPASS) {
        static string gl_pass;
        cin >> gl_pass;
        return (char*)gl_pass.c_str();
    }
    else if (mode == PDFFA_OUT_FILENAME) {
        if(user_data != 0) {
            //stores the filename in the given vector
            vector<string>* saved_files=(vector<string>*)user_data;
            saved_files->push_back(string(msg));
        }
    }
    return 0;
}
```

```
' or in VB.NET...
Public Function MyCallback(ByVal mode As Integer, ByVal msg As String,
ByVal user data As Int32) As Int32
```

```

If mode = PDF2IMAGE_ERROR Then
    Console.WriteLine("Error: {0}", msg)
ElseIf mode = PDF2IMAGE_MSG Then
    Console.WriteLine("{0}", msg)
ElseIf mode = PDFA_OUT_FILENAME Then
    If user_data.ToInt32 <> 0 Then
        'add the next saved filename to the given ArrayList
        Dim gch As GCHandle = GCHandle.op_Explicit(user_data)
        Dim list As ArrayList = CType(gch.Target, ArrayList)
        list.Add(msg)
    End If
End If
Return 0
End Function      'MyCallback

```

Please note that, by default, PDF/A Manager SDK will also generate 'report.xml' outlining validation and conversion results for each file. To disable report generation simply include '--noxml' in the command-line string.

5.3 PDF/A Add-on module for PDFNet SDK

All of functionality available in PDF/A Manager SDK is available as an add-on module to PDFNet SDK (<http://www.pdftron.com/pdfnet>). PDFNet SDK is an amazingly comprehensive, high-quality PDF developer toolkit for working with PDF files at all levels. Using the PDFNet PDF library, developers can flexibly implement and create powerful PDF solutions and applications that can generate, manipulate, view, render and print PDF documents without any third-party software dependencies.

PDFNet SDK is available as a .NET component and as a cross-platform Java and C/C++ PDF library available on a wide range of platforms.

This section only illustrates the basic use of the PDFACompliance class in PDFNet SDK. To obtain the full sample code or PDFNet API Reference, you can simply download PDFNet SDK for your development platform from: <http://www.pdftron.com/pdfnet/downloads.html>.

```

// C# sample code (C/C++, JAVA, and VB6 samples are also available).

using System;
using pdftron;
using pdftron.PDF;
using pdftron.PDF.PDFA;

namespace PDFATestCS
{
    class Class1 {
        // Relative path to the folder containing test files.
        static string input_path = "../..../TestFiles/";
        static string output_path = "../..../TestFiles/Output/";

        static void Main(string[] args)
        {
            PDFNet.Initialize();
            PDFNet.SetColorManagement(true); // Required for PDFA validation.

            // Example 1: PDF/A Validation

```

```

try {
    string filename = "newsletter.pdf";
    PDFACompliance pdf_a = new PDFACompliance(false,
input_path+filename, null, PDFACompliance.Conformance.e_Level1B,
null, 10, false);
    PrintResults(pdf_a, filename);
    pdf_a.Dispose();
}
catch (pdftron.Common.PDFNetException e) {
    Console.WriteLine(e.Message);
}

// Example 2: PDF/A Conversion
try {
    string filename = "fish.pdf";
    PDFACompliance pdf_a = new PDFACompliance(true,
input_path+filename, null, PDFACompliance.Conformance.e_Level1B,
null, 10, false);
    filename = output_path + "pdfa.pdf";
    pdf_a.SaveAs(filename, true);
    pdf_a.Dispose();

    // Re-validate the document after the conversion...
    pdf_a = new PDFACompliance(false, filename, null,
PDFACompliance.Conformance.e_Level1B, null, 10, false);
    PrintResults(pdf_a, filename);
    pdf_a.Dispose();
}
catch (pdftron.Common.PDFNetException e) {
    Console.WriteLine(e.Message);
}
PDFNet.Terminate();
}

static void PrintResults(PDFACompliance pdf_a, String filename) {
    int err_cnt = pdf_a.GetErrorCount();
    if (err_cnt == 0)
        Console.WriteLine("{0}: OK.\n", filename);
    else {
        Console.WriteLine("{0} is NOT a valid PDF/A.", filename);
        for (int i=0; i<err_cnt; ++i) {
            PDFACompliance.ErrorCode c = pdf_a.GetError(i);
            Console.WriteLine(" - e_PDFA{0}: {1}.",
                c,
                PDFACompliance.GetPDFAEErrorMessage(c));
        }
    }
}
}

```

6. Support

6.1 Reporting Problems

If you encounter a problem or question regarding PDFTron's PDF/A Manager, which is not addressed on PDFTron's website, please submit a problem report to PDFTron's Support group at <http://www.pdftron.com/support/reportproblem.html>.

When submitting a problem you will be asked to provide the following information:

- Contact details
- Product and Version of the product
- Detailed description of problem
- Problem file(s)
- Whether you have an AMS (Annual Maintenance Subscription)
- Any other information that may be related

6.2 Contact Information

To contact PDFTron directly, please use the contact information below:

Tel: 1-604-730-8989

Fax: 1-604-676-2477

Web site: www.pdftron.com

Email Contacts:

General Business Inquiries: info@pdftron.com

Sales & Licensing: sales@pdftron.com

Product Support: support@pdftron.com

Professional Services: services@pdftron.com

Website related questions: webmaster@pdftron.com

Press & News: press@pdftron.com