Morovia PDFLeo Manual

Morovia Corporation

PDFLeo 1.0 User Manual

PDFLeo 1.0 User Manual

Publication date: October 2010
Revision: 4202

Technical Support
Phone: (905) 752-0226
Fax: (905) 752-0355
Email: support@morovia.com
Web: http://www.morovia.com

For more information about Morovia products, visit http://www.morovia.com.

Table of Contents

List of Tables

Chapter 1. Introduction

PDFLeo is a command line utility to transform existing PDF files - such as linearization, encryption, decryption, compression, size reduction, and metadata modification. It can also be used to display information of PDF documents, such as metadata, security attributes and required fonts.

A typical use of PDFLeo is to post-process PDF files created by a word processor or a PDF generator, such as Word 2007 or FOP. You can use this program to optimize (linearization or size reduction), encrypt, add metadata etc. In addition to modifying existing files, you can use the program to collect meta data of PDF documents, such as title, author and creation date.

Both 32-bit and 64-bit executables are included in the package.

1.1. Editions

PDFLeo comes with three editions - Standard, Professional and Server. The first two editions can't run on Windows server operating systems, such as Windows 2003 Server or 2008 Server.

License does not restrict the number of CPUs that the program runs.

- The Standard Edition includes all the features except reading and writing public key encrypted PDF files. Standard edition can't run on Windows servers.
- The Professional Edition has all the features enabled, including reading and writing public key encrypted PDF files. However, it can't run on Window servers.
- The Server Edition has all the features enabled, including public key encryption/decryption support. It can run on both server and workstation Windows systems.

In addition to the three full editions, a trial version is available to download from our web site. The trial version contains all the features, and run on both server and workstation Windows, except that it can only process PDF files up to 50 pages. The trial version is supplied for evaluation purpose and must not be used for production purposes.

As of the writing, PDFLeo requires Windows 2000 or above. For availability on other platforms, send your inquiry to info@morovia.com.

1.2. Installing the Software

The software is packed in a zip file. Unzip it to your Windows folder or a new folder created. Optionally you may add the directory to your PATH variable so that you can call from other directories.

All editions of PDFLeo, including the trial version, share the same binary. After your purchased the software, you will receive a license key that activates the edition you bought. To convert the trial version into the edition you bought, fill in pdfleo.lic file with the license key received. The contents of the file look like below:

```
<?xml version="1.0" encoding="UTF-8"?>
   <license>
       <product>pdfleo</product>
       <version>1.0</version>
       <licensee>ABC Corporation</licensee>
       <serialnumber>E722J-UCKWM-QRAD2-48Q3T-IZCNRE</serialnumber>
   </license>
```

In most cases, both licensee and serial number fields must match the ones in the e-mail or on the package cover. PDFLeo will read and validate the license code at startup. If validation failed, pdfleo runs under trial mode, which processes PDF files up to 50 pages.

Note The `pdfleo.lic` file must be a valid XML file in UTF-8 encoding. If licensee name contains non-ASCII characters, use a UTF-8 capable text editor to modify the contents.

After installing the software, run the command `pdfleo --help` to verify that the software is installed correctly. If `license name` and `serial number` were correctly entered, the program will print the edition at the first line.

```
C:\pdfleo --help
Morovia (R) pdfleo Professional❶ 64-bit❷ Version 1.0 (build 4104)❸
Usage:  pdfleo [-p|--password=<pwd>]... [--digital-id=file:pwd]...
...
```

❶ Professional Edition
❷ 64-bit executable
❸ Software version 1.0 build number 4104

If you need to test large PDF files, write to support@morovia.com to request time based trial keys. Keys will be granted case by case.

Chapter 2. Overview

Portable Document Format (PDF) is a file format created by Adobe Systems for document exchange. Each Adobe PDF file encapsulates a complete description of a fixed-layout 2D document that includes the text, fonts, images, and 2D vector graphics which compose the documents. PDF is an ISO standard, published under title *Document management—Portable document format—Part 1: PDF 1.7*.

PDFLeo allows users to post-process PDF files generated by other programs, such as PDF printers, word processors (such as Word 2007 and OpenOffice Writer), and other programs such as FOP and iText. Many of these programs lack some features. For example, Some does not provide encryption. Some produce files with quite large sizes. And some leave metadata fields empty. Most of them do not produce web optimized files. Those shortcomings can be fixed with pdfleo.

2.1. Product Features

PDFLeo supports the following kinds of PDF processing:

- Encryption. Encrypt a PDF document with either password security or public key security. Remove PDF encryption if you are able to open the file. Retain PDF encryption and permission settings in the new PDF with other part of document modified.
- Linearization. Optimize PDF documents to be viewed over slow connection from a capable web server.
- Size Optimization. Reduce the file size by removing redundant contents, compressing streams and moving objects to streams.
- Query document information, such as meta data, security, document permission and font information.
- Insert and modify predefined or custom document information entries.
- Insert, view and modify XMP metadata.

2.2. Technical View

A PDF file consists primarily of objects, of which there are eight types:

- Boolean values, representing true or false
- Numbers
- Strings
- Names
- Arrays, ordered collections of objects
- Dictionaries, collections of objects indexed by Names
- Streams, usually containing large amounts of data
- The null object

Objects may be either direct (embedded in another object) or indirect. Indirect objects are numbered with an object number and a generation number. An index table called the xref table gives the byte offset of each indirect object from the start of the file. This design allows for efficient random access to the objects in the file, and also allows for small changes to be made without rewriting the entire file (incremental update).

Beginning with PDF version 1.5, indirect objects may also be located in special streams known as object streams. This technique reduces the size of files that have large numbers of small indirect objects.

2.2.1. Layouts

There are two layouts to the PDF files: normal and linearized. Non-linearized PDF files consume less disk space than their linear counterparts, though they are slower to access because portions of the data required to assemble pages of the document are scattered throughout the PDF file. Linear PDF files (also called "optimized" or "web optimized" PDF files) are constructed in a manner that enables them to be read in a Web browser plugin without waiting for the entire file to download, since they are written to disk in a linear (as in page order) fashion.

2.2.2. Metadata

Metadata is specified in two ways: the "native" info dictionary, and the metadata stream. The info dictionary method suffers several shortcomings: Unless the indexing software understands the format, it is not possible to locate the metadata. Adobe only published a few metadata entries, and there is no standard method on the names and contents. Furthermore, only a string per entry is supported in the info dictionary. If the file is encrypted, the index software requires the key to access the entries.

PDF metadata stream stores metadata in a standard format called XMP. XMP is serialized and stored using W3C RDP format (which is based on XML), and can be placed into a variety of multimedia file such JPEG. When stored as plain text in the file, XMP contains a special header indicating that it is XMP as well as the encoding format. Therefore, an indexing software can search the file for this special mark, and loads the content without understanding the PDF file format.

PDFLeo allows user to insert XMP metadata into existing PDF files, as well as modifying existing info dictionary entries. When info dictionary is modified, the change is synced back to the metada.

2.2.3. Encryption

PDF allows all string and stream objects (except metadata stream) to be encrypted to prevent unauthorized people to access the content. The encryption method employed can be either RC4 (with key length from 40 bits to 128 bits) and AES (128 bits). Before decrypting the content, a master key must be obtained. Security handler is to calculate the master key. PDF standard defines two kinds of security handlers, and pdfleo supports both.

- Password security handler. Called standard handler in PDF specification, this security handler allows user to specify two passwords: the user password, and the owner password. Through some data stored in the PDF file, the user password can be calculated once the owner password is known. The user password is used to decrypt the master key.
- Public key security handler. This security handler uses X509 public certificate to encrypt the master key. Multiple receipts can be specified in this manner. The user supplies his private key when he views the file. In this manner no password exchange is required. The creator only needs the public certificate of the recipient.

It is possible that a PDF uses a non-documented security handler, or uses a live server for authentication. pdfleo can't decrypt these files, as most other PDF programs.

2.3. Supplying Credentials

If the source document is encrypted, credentials are required to gain access to the document. PDF standard defines two types of securities - password and public key. There are other types of possible securities, but they are not standard therefore can not be read by pdfleo.

2.3.1. Specifying Passwords

Password security allows two passwords - the owner password and the user password. Users with owner passwords have unrestricted privileges to the document, while those with user passwords have limited permissions defined by the author. It should be noted that the permission is enforced by the application. Users with user password effectively have unrestricted access to the document with the help of the application. For example, a PDF document with empty user password can have the encryption settings removed with pdfleo.

Passwords are specified through `--password` switch. If password is empty, there is no need to specify as `pdfleo` will try using empty password if other attempts failed.

Multiple passwords, up to 10 can be specified at the command line. pdfleo will try them in the order specified. pdfleo does not honor the permission flag, specifying either password will work in most cases.

```
C:\>pdfleo --password=oooo134 --password=sjusl \
    source.pdf target.pdf
```

PDFLeo will try passwords in the order that they are specified. Some action, such as copying owner password to the new file, requires the file opened by an owner password. Therefore, if both passwords are known, owner password should be place before user password.

2.3.2. Specifying digital ID file

Public key encrypted PDF files require private key file and the password to access the private key file. A colon (:) separates the two parts. The private key file must be in standard pkcs#12 format and usually has extension of `pvk` or `p12`. If password is required to access the private key, it must be specified. For example, the following parameter identify private key file `c:\company.pvk` and the password is `password`.

```
C:\>pdfleo --digital-id=c:\company.pvk:mypass source.pdf target.pdf
```

Multiple digital ID switches, up to 10 can exist at the command line. PDFLeo will try them in the order that they are specified.

2.4. Encryption, Decryption and Permissions

The detailed information on how to encrypt, decrypt or change permissions is covered in Chapter 3, *PDF Security*. This section gives a quick summary and some examples.

2.4.1. Querying Security Settings

Security settings are displayed using --info switch.

```
C:\>pdfleo -i test2.pdf
...
=============== Document Security ==============================
Security Method: Password Security ❶
  Authorized by: User Password     ❷
          Print: None              ❸
  Accessibility: No
         Modify: Assembly
Encryption Level: AES (128-bit)    ❹
...
```

❶ Document is encrypted with passwords
❷ Document is opened using user password (which is empty)
❸ Print permission: no print allowed
❹ Encryption method is 128-bit AES.

For public key encrypted documents, the program prints list of recipients:

```
C:\>pdfleo --digital-id=tester.pfx:1234demo pubkey.pdf
...
=============== Document Security ==============================
Security Method: Certificate Security
          Print: High Resolution
  Accessibility: No
         Modify: Assembly
 Clear Metadata: No
Encryption Level: AES (128-bit)

List of Recipients:
C=US, O=Equifax, OU=Equifax Secure Certificate Authority ❶
CN=Test Group, O=Morovia, OU=Morovia/emailAddress=tester@morovia.com, C=CA ❷
...
```

❶ Recipient #1: Equifx Certifcate Authority
❷ Recipient #2: Test Group (tester@morovia.com)

2.4.2. Encrypting Document

PDF documents can be encrypted with password security or public key security. A number of parameters can be specified such as encryption level, bit length and so on.

```
C:\>pdfleo --encrypt=new;AES-128;;yes \ ❶
   --password-security=demo;;print=high;modify=none  ❷
```

❶ Document is encrypted with 128-bit AES. Metadata is not encrypted.
❷ Document is secured with passwords - owner password is demo. User password is empty. Permission: high resolution and now modification.

2.4.3. Removing Encryption

Encryption is removed by specifying `--encrypt=discard` without additional parameters.

```
C:\>pdfleo --password=Monster --encrypt=discard \
```

```
    source.pdf target.pdf
```

2.4.4. Preserving Encryption

By default, the encryption is preserved in the output PDF. All security setting remain intact. For example, the below command creates a linearized PDF file with encryption copied from the source PDF:

```
C:\>pdfleo --password=Monster \
    --linearize source.pdf target.pdf
```

The preserve mode can be explicitly specified though `--encrypt=preserve` parameter. The command below achieves the same results:

```
C:\>pdfleo --password=Monster --encrypt=preserve\
    --linearize source.pdf target.pdf
```

2.5. Web-Optimized (Linearized) PDF

A non-linearized PDF file must be fully downloaded to the client computer before a viewer can display the pages. Linearization (called *Fast Web View* in Acrobat and Adobe Reader) transforms PDF into such a format that a capable viewer can find out which byte range to ask for when display the page requested with only a few K bytes downloaded. It then asks web server for bytes within that rage. The capability of "byte-serving" is required by HTTP 1.1 protocol and is supported by most web servers.

In order to take advantage this feature, you need:

- A web server that supports byteserving. As the protocol is part of http 1.1, most current web servers already are capable.
- The viewer must support this feature. In Acrobat or Adobe Viewer, make sure that the option Allow fast web view is checked. This option is enabled by default.
- The PDF file is linearized, which can be achieved by pdfleo.

Linearization provides better user experience when serving large PDF files (measured in number of pages or file size in MB) over web or other slow connections. Linearization will generally increase the file size. In some cases it increases the file size significantly if you have a large PDF file and many objects are compressed into an object stream.

Linearization feature is applied through `-linearize` switch. It can be used in conjunction with other transforms, such as encryption and compression.

```
C:\>pdfleo --linearize source.pdf target.pdf
```

You can verify the result using `pdfleo --info`. As illustrated below:

```
C:\>pdfleo --info target.pdf
  ...
Number of Pages: 30
     Tagged PDF: No
     Linearized: Yes ❶
      Page Size: 8.50x11.00 in
  ...
```

❶ the document is linearized

You can also verify it through Adobe Reader or Acrobat. Open the document and select File → Properties... (**Ctrl+D**). The `Fast Web View` entry will show `Yes`, which indicates that the document is linearized.

Note Linearization is *not* preserved during transformation, unless `--linearize` option is specified. If the source PDF is linearized but no `--linearize` is specified, the resulted PDF will not be linearized.

2.6. Optimization (Size Reduction)

Due to the method that pdfleo utilizes to read the source document, some optimization techniques are always applied. Additional steps can be taken to further reduce file size, such as compressing stream objects and placing objects into streams. The techniques involved include:

- Removing unused objects. Unused objects will be discarded. If a PDF is produced through incremental update, many objects are not needed. Incremental update is a feature to allow a processing application to append changes at the file end without removing prior object definitions. This technique reduces the memory usage at the cost of bigger file size.
- Writing objects in a compact syntax. PDFLeo writes output using compact syntax Extra white spaces are removed. Hexadecimal strings are written with more compact binary representations.
- Compressed streams. When specified, pdfleo compresses all streams except those who must be kept intact.
- Object streams. Non stream objects can be placed into a special object stream and compressed.

Optimization is controlled through two switches: `--stream-data` and `--object-stream`. The `--stream-data` option controls how individual stream is compressed. `--object-stream` controls whether or not object streams should be generated.

PDFLeo does not apply any optimization steps which could result loss of information such as image quality degradation, loss of fonts etc.

2.6.1. Stream Compression Options

Contents of stream objects can be encoded with various techniques, such as LZW compression or Flate compression. The techniques are referred as *filters*. An application program that produces a PDF file can encode certain information (for example, data for sampled images) to compress it or to convert it to a portable ASCII representation. Then an application that reads (consumes) the PDF file can invoke the corresponding decoding filter to convert the information back to its original form.

stream compression options is specified through `--stream-data` switches and can be one of three values: `none`, `preserve` or `compress`.

Table 2.1. Stream Compression Options

Value	Description
none	With this option all streams are uncompressed. This is useful if you want to look at the plain text content.
preserve	Preserve the state of current streams. i.e., if the original stream is compressed, the output PDF is also compressed.
compress	Apply deflate compression on all data streams, unless application could cause adverse effects.

PDFLeo understands many filters. However, in terms of compression, Flate filter is utilized exclusively (another compression filter is LZW, which does not offer better performance). However, pdfleo will preserve stream data when it encounters an unknown filter or a loss filter such as DCTDecode. For streams with predictor, it will decompress as required, but will preserve them at compression.

Metadata streams are handled differently in pdfleo. In order for metadata is search able, the stream should be in clear text (uncompressed and unencrypted). Therefore, PDFLeo does not compress metadata streams even `--stream-data=compress` is specified. This behavior happens when the PDF document is either

unencrypted, or encrypted with clear text metadata. If metadata is encrypted, metadata streams will also be compressed.

The command below compresses data streams to reduce file size:

```
C:\>pdfleo --data-streams=compress source.pdf target.pdf
```

Occasionally some users may want the data streams in clear text, so that they can see the drawing commands in each page. The following command line produces such output.

```
C:\>pdfleo --data-streams=uncompress source.pdf target.pdf
```

2.6.2. Object Stream Options

PDF file size can be substantially reduced by placing uncompressed PDF objects into a stream and compressing the stream. This type of stream is called *Object Stream*. Object streams option is specified through `--object-streams`, and can be one of the three values below:

Table 2.2. Object Stream Options

Value	Description
preserve	Original object streams are preserved. If the source document does not contain object streams they are not generated.
disable	Removing object streams. This often results a PDF file with bigger size.
generate	Generating object streams whenever possible.

The following command compresses all data streams, and placing object into object streams whenever possible, with the goal to minimize the file size.

```
C:\>pdfleo --data-streams=compress \
   --object-streams=generate source.pdf target.pdf
```

The default option is `preserve`.

2.7. Querying Document Information

You can obtain various of metadata of a PDF document by using option `--info`, or abbreviated option `-i`. The output first lists all information dictionary entries, followed by document security attributes such as security method and permission. The last section lists all the fonts required by the document (either embedded or required to be present in the system).

Below is the sample output:

```
C:\pdfleo --info Brother_HL_4050_CDN_Manual.pdf
Morovia (R) pdfleo 32-bit Professional Version 1.0
           File: Brother_HL_4050_CDN_Manual.pdf
          Title: HL4040CN_HL4050CDN_HL4070CDW.book ❶
         Author: ZZPZ3635
        Subject: N/A
       Keywords: N/A
        Created: 06/29/2007 10:38:30 AM
       Modified: 06/29/2007 04:05:36 PM
    Application: FrameMaker 7.0
   PDF Producer: Acrobat Distiller 6.0 (Windows)
    PDF Version: 1.5 (Acrobat 6.x)
Number of Pages: 211
     Tagged PDF: No
     Linearized: Yes
      Page Size: 8.50x11.00 in
================ Document Security ================================
Security Method: Password Security               ❷
  Authorized by: User Password
          Print: Allowed
         Modify: Not Allowed
        Extract: Allowed
       Annotate: Not Allowed
Encryption Level: RC4 (40-bit)

================ Fonts Info ======================================
Font Name                                Encoding     Type
---------------------------------------- ------------ ------------
TT9A1o00(embedded subset)                WinAnsi      Type 1C ❸
TT9A2o00(embedded subset)                WinAnsi      Type 1C
TT9A3o00(embedded subset)                WinAnsi      Type 1C
TT9A4o00(embedded subset)                WinAnsi      Type 1C
TT9A5o00(embedded subset)                WinAnsi      Type 1C
 ... (remaining skipped)
```

❶ Document metadata
❷ Security properties
❸ List of document fonts

2.8. Inserting and Modifying Info Dictionary

PDF stores metadata entries in a special dictionary, called *information dictionary*. PDF defines several standard keys, and authors can define custom entries. Often info dictionary is called *native*. Because the limitations of the dictionary, multiple language values are not supported.

Despite the efforts to push XMP adoption, many PDF software read the info dictionary for metadata. Therefore, it is necessary to keep the info dictionary and XMP metadata in synchronization.

PDFLeo allows users to insert, modify or remove entries in the information dictionary. Changes will synchronize to XMP metadata.

The following command snippet changes the document title to **`PDFLeo User Manual`**, document producer to **`Morovia PDF Writer`**. If the specified key does not exist, it will be added.

```
--info-dict="Title=PDFLeo User Manual;Producer=Morovia PDF Writer"
```

By specifying empty value you can remove an entry. For example,

```
--info-dict="Title=;Producer:Morovia PDF Writer"
```

removes the `Title` entry.

2.8.1. Standard Information Dictionary Entries

Standard entries in the dictionary are listed as below:

Table 2.3. Entries in the document information dictionary

Key	Value
Title	The document's title
Author	The name of the person who created the document.
Subject	The subject of the document
Keywords	Keywords associated with the document.
Creator	The name of the application that created the original text (such as Microsoft Word).
Producer	The name of the application that converted the original document into the PDF format (such as PDF printer)
CreationDate	The date and time that the document is created.
ModDate	The date and time that the document is modified.
Trapped	A value indicating the document has been modified to include trapping information. Valid values include: True, False and Unknown.

Note that the keys are case sensitive. For entries requiring a Date (such as `ModDate` and `CreationDate`), the value must conform to the date format defined in the PDF standard, which you can find in Section A.1, “PDF Date Format”.

2.9. XMP Metadata Processing

XMP (Extensible Metadata Platform1) is an XML framework with many predefined properties. However, as the name implies, XMP can be extended to satisfy specific requirements using custom extension schema. XMP is much more powerful than document information dictionary, and is for example required in the PDF/A standard. Many industry groups have published standards based on XMP for various vertical applications, e.g. digital imaging or prepress data exchange.

With `pdfleo` you can extract duocument level XMP metadata, replace or merge existing metadata with contents from an external file.

2.9.1. Extracting XMP

XMP can be extracted through `--dump-xmp` switch. The dump content is expressed in UTF-8 format.

Figure below shows the results of running pdfleo with `--dump-xmp` option on the `xmpguide.pdf`. Note that the metadata contains five schema: PDF schema (http://ns.adobe.com/pdf/1.3/", XMP Basic Schema (http://ns.adobe.com/xap/1.0/"), Dublic Core Schema (http://purl.org/dc/elements/1.1/"), XMP PDFX Schema (http://ns.adobe.com/pdfx/1.3/") and XMP Media Management Schema (http://ns.adobe.com/xap/1.0/mm/").

```
<x:xmpmeta xmlns:x="adobe:ns:meta/" x:xmptk="Adobe XMP Core 4.0-c316 44.253921, Sun Oct 01 2006 17:14:39">
   <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
      <rdf:Description rdf:about=""
            xmlns:pdf="http://ns.adobe.com/pdf/1.3/">
         <pdf:Copyright>Copyright 2010, Adobe Systems Incorporated, all rights reserved.</pdf:Copyright>
         <pdf:Producer>Acrobat Distiller 8.1.0 (Windows)</pdf:Producer>
         <pdf:Keywords>XMP metadata, XMP Toolkit,  SDK, XMP File Handler, XMP Core API</pdf:Keywords>
         <pdf:Marked>True</pdf:Marked>
      </rdf:Description>
      <rdf:Description rdf:about=""
            xmlns:xap="http://ns.adobe.com/xap/1.0/">
         <xap:ModifyDate>2010-04-12T09:29:35-07:00</xap:ModifyDate>
         <xap:CreatorTool>FrameMaker 8.0</xap:CreatorTool>
         <xap:CreateDate>2010-04-12T08:58:24Z</xap:CreateDate>
         <xap:MetadataDate>2010-04-12T09:29:35-07:00</xap:MetadataDate>
      </rdf:Description>
      <rdf:Description rdf:about=""
            xmlns:dc="http://purl.org/dc/elements/1.1/">
         <dc:format>application/pdf</dc:format>
         <dc:title>
            <rdf:Alt>
               <rdf:li xml:lang="x-default">XMP Toolkit SDK Programmer's Guide</rdf:li>
            </rdf:Alt>
         </dc:title>
         <dc:creator>
            <rdf:Seq>
               <rdf:li>Adobe Developer Technologies</rdf:li>
            </rdf:Seq>
         </dc:creator>
         <dc:description>
            <rdf:Alt>
               <rdf:li xml:lang="x-default">Using the XMP Toolkit  SDK to work with  XMP metadata</rdf:li>
            </rdf:Alt>
         </dc:description>
      </rdf:Description>
      <rdf:Description rdf:about=""
            xmlns:pdfx="http://ns.adobe.com/pdfx/1.3/">
         <pdfx:Copyright>Copyright 2010, Adobe Systems Incorporated, all rights reserved.</pdfx:Copyright>
         <pdfx:Marked>True</pdfx:Marked>
      </rdf:Description>
      <rdf:Description rdf:about=""
            xmlns:xapMM="http://ns.adobe.com/xap/1.0/mm/">
         <xapMM:DocumentID>uuid:60929191-890d-445c-be1f-645ff48fe9a9</xapMM:DocumentID>
         <xapMM:InstanceID>uuid:5b913814-3ecb-4841-9bd7-c56ee2ffdc20</xapMM:InstanceID>
      </rdf:Description>
   </rdf:RDF>
</x:xmpmeta>
```

PDF Schema

XMP Basic Schema

Dublin Core Schema

PDFX Schema

XMP Media mgmt Schema

PDFLeo provides another switch, `--dump-xmp-text` to display metadata in a more readable format:

```
c:\>pdfleo --dump-xmp-text xmpspec3.pdf
dumping metadata:Dumping XMPMeta object ""  (0x0)
```

```
pdf:  http://ns.adobe.com/pdf/1.3/  (0x80000000 : schema)
   pdf:Copyright = "Copyright 2010, Adobe Systems ...
   pdf:Marked = "True"
   pdf:Producer = "Acrobat Distiller 8.1.0 (Windows)"
   pdf:Keywords = "XMP metadata  EXIF TIFF IPTC PSIR"
   ...
```

2.9.2. Replacing XMP metadata

You can replace existing XMP metadata with contents of an external file by using `--replace-xmp` switch. The file must be a valid XML file with UTF-8 encoding.

The following example replaces metadata with contents from a file named `ticks.xmp`:

```
C:\pdfleo --replace-xmp=ticks.xmp source.pdf output.pdf
```

Note that you should make sure that the content is correct. The file should be coded with UTF-8. And the content of metadata is not synchronized to the info dictionaries.

2.9.3. Merging XMP metadata

PDFLeo provides another switch, `--merge-xmp` to allow users to merge contents of an external file with the existing metadata. New properties are added, and old properties are replaced.

The following example updates metadata with contents from a file named `ticks.xmp`:

```
C:\pdfleo --merge-xmp=ticks.xmp source.pdf output.pdf
```

Note that you should make sure that the content is correct. The file should be coded with UTF-8. And the content of metadata is not synchronized to the info dictionaries.

Chapter 3. PDF Security

A PDF file can be encrypted to prevent access of unauthorized users. When a PDF file is encrypted, all string and stream objects are encrypted using a symmetric-key algorithm, either RC4 or AES. PDF starts with RC4 40 bit at version 1.2, and at version 1.6, it supports 40-bit RC4, 128-bit RC4 and 128-bit AES. The encryption method and key length combined is referred as *Encryption Level*. Symmetric-key encryption uses the same key to encode and decode data, often called *File Key* or *Master Key*.

The access method to the file key is available in a number of ways, called *Security Handler*. Two security handlers are defined in the PDF standard: *password security* and *public key security* (also called *certificate security*). There are other proprietary security handlers available, such as Adobe Policy Server security, which requires a live proprietary server at presence. Due to obvious reasons, PDFLeo does not support those proprietary handlers.

PDF standard also defines access permission which a document author can specify the desired access of users with certain credentials when encryption is applied. Unfortunately, access permission is not enforced at cryptographic level. Once a PDF is decrypted, the program has access to all the document objects. It is up to the client program to honor the permission.

The permission settings varies by PDF version. The complete list can be found at Section 3.3.2, “Permissions”.

The metadata stream in the document can be optionally marked as clear text, in order for other software (such as a search engine) to retrieve the data without requiring credentials. Clear text metadata support is available at PDF version 1.5, and requires 128-bit encryption level (RC4-128 or AES).

3.1. Compatibility Levels

Encryption setting selection may affect the PDF version. As indicated previously, pdfleo automatically increases PDF version of the output file if a feature selected requires so.

Occasionally user may want to limit the PDF version, in order to ensure that results are compatible with existing software. In order to do so user needs to know which encryption settings are supported at that PDF version. This manual groups encryption settings into *Compatibility Level*.

A higher PDF version can utilize lower compatibility level. For example a PDF file marked version 1.6 can be encrypted with 40-bit RC4 (requiring 1.2). However under this settings metadata stream can't be marked as clear text. In order for the metadata to be marked as clear text, 128-bit RC4 or AES must be used.

To select encryption options based on specified PDF version, first determine the highest compatibility level. For example, suppose that the resulted PDF should have a version no higher than 1.5 (Acrobat 6.0). From the list below, 128-bit AES can't be selected, but 40-bit RC4 (with metadata always encrypted) is a candidate.

- Acrobat 3.0 (PDF version 1.2). Encryption level is 40-bit RC4. Metadata is always encrypted.
- Acrobat 5.0 (PDF version 1.4). 128bit-RC4. Metadata is always encrypted.
- Acrobat 6.0 (PDF version 1.5). 128bit-RC4. Metadata can be marked as clear text.
- Acrobat 7.0 (PDF version 1.6). 128bit-AES Metadata can be marked as clear text.

Warning PDFLeo does not currently support AES-256 encryption settings per PDF 1.7 Adobe Extension Level 3/ Acrobat 9.

3.2. Security Handlers

3.2.1. Password Security

Password security features two passwords: *user password*, which is given to the viewer of the document, and *owner password*, which is reserved by the author. In theory, without owner password, the user can only perform operations allowed by the author, such as printing, extracting pages and so on. The author, while retaining the owner passwords, has the full access to the document. However, as pointed out previously, the permission is only enforced by the application, not by cryptographic technology. Anyone with user password can convert the document to unencrypted mode. Some authors choose to set user password to blank and distribute the documents with expectation that anybody can view the document, but can't modify it. This expectation is not correct.

3.2.2. Public Key Security

Although password security protects the encryption key to authorized users, the passwords must be known to authorized users prior to viewing the document. In many times it is difficult to pass the password in a secured manner. public key security solves the problem in a nice way - the authorized users give their certificates to the author, and encryption key is encrypted with the certificate. The certificate can be passed over unsecured media because the decryption requires the private key, which is not transferred.

The two security handlers are mutually exclusive. If public key security is chosen, the document can't be encrypted with a password, and vice versa.

Similar to the password security, the author can set permission for each user, referred as recipient. Multiple recipients can share the same permission, called recipient set. A PDF document can have multiple permission set. Like passwords security, the permission is only enforced by the application.

3.3. Specifying Encryption Settings

PDFLeo supports two types of security handlers - password security and public key security. Both share common encryption settings - such as encryption level, key length and if metadata is encrypted. The common settings are specified through `--encrypt` switch. After this is specified, you use `--password-security` to specify parameters for password security, or --recipient to specify parameters for public key security.

3.3.1. Common Settings

Common settings are specified through `--encrypt` switch. The value of this option has four fields, divided by semicolon:

1. The first field is one of the three choices indicating how the current encryption setting is based: preserve, copy, and discard.

   preserve

   This is the default option when no `--encrypt` is present. The output PDF preserves all encryption settings, including passwords, permission and recipient sets. If the original PDF file is not encrypted, the output will not be encrypted either. Under this mode, no `--password-security` or `--pubkey-recipient` is allowed. The switch works as long as you have the credentials to open the input file.

   discard

   The output PDF will use encryption settings specified at the command line, without any inheritance from source file. Normally it is followed by encryption level, key length and metadata

encryption flag. However, if none of them is present, the resulted PDF will have no encryption at all.

copy

The encryption setting is copied from source document, and subsequently modified by options specified at command line. The attributes copied include encryption level, key length, and permissions. Note that this option is used to establish the base line for encryption settings. They should not be used when the whole encryption attributes are preserved in the target document.

2. Encryption Level: can be RC4, RC4-128, and AES-128. Encryption level affects the following key length parameter as some of them have fixed key length.
3. Key Length. For RC4-128 encryption level, key length between 40 and 128 can be specified here. Note that the key length must be integral times of 8. For other encryption levels, this field is ignored. For RC4 cipher the key length is fixed at 40 bits. For AES-128, the key length is 128 bits.
4. A word `yes` or `no` to indicate if meta data stream should be kept clear text. This setting is available when encryption level is RC4-128 or AES-128. The default value is `yes`.

Note The difference between `copy` and `discard` is the encryption setting baseline. If `copy` is specified, the following settings are copied from the source PDF: encryption leve, key length, clear text metadata and permissions. If `discard` is specified, the default base line is in use: AES-128, clear text metadata, and full access permission. Subsequent parameters will modify those settings. If the source is not encrypted, the two modes produce the same result.

3.3.2. Permissions

Permission is specified inside `--password-security` or `--pubkey-security`. For public key encrypted PDFs, recipients are grouped into recipient set, with each set sharing the same permission.

Note that permission settings are based on encryption level.

If RC4 (40-bit) is specified, the following permissions are available.

print=[yes|no]

Determines whether or not to allow printing.

modify=[yes|no]

Determines whether or not to allow document modification.

extract=[yes|no]

Determines whether or not to allow text/image extraction.

annotate=[yes|no]

Determines whether or not to allow comments and form fill-in and signing.

If RC4-128 or AES-128 is specified as encryption level, the following permissions are applicable:

accessibility=[yes|no]

If yes, text access for screen reader devices for the visually impaired is allowed.

extract=[yes|no]

If yes, text and graphic extraction, such as copying of text, images and other content, is allowed.

print=[high|low|none]

This option specifies the print access, which can be one of the following:

- `high`: allow high resolution printing.

- `low`: allow degraded printing.
- `none`: no printing is allowed.

modify=[all|annotate|form|assembly|none]

This option specifies the modify access, which can be one of the following, each of which implies all the options that follow it:

- `all`: allow full document modification
- `annotate`: allow comment authoring and form operations
- `form`: allow form field fill-in and signing existing signature fields.
- `assembly`: allow document assembly only (inserting, deleting and rotating pages).
- `none`: allow no modifications.

Permissions are separated with semicolon. For example, the following code specifies permission that allow high resolution printing and no modification:

```
C:\>pdfleo --encrypt=copy; --password-security=pdf123;;print=high;modify=none
```

3.3.3. Password Security Parameters

Use `--password-security` switch to specify owner password, user password and permission. This switch can't be used with `--encrypt=preserve`.

The value of switch requires at least two fields - the first field containing the value of owner password, followed by user password. Permissions are optional, and start from the third field if they are present. Fields are separated by semicolon.

```
--password-security=<ownerpass>;<userpass>;<perm1>;<perm2>...
```

Empty passwords are allowed by leaving the field blank. You can also copy the password from the source file using asterisk (*). If source document is not encrypted, empty password is used for the output. For example,

```
C:\>pdfleo --encrypt=copy --password-security=pdf123;*;print=high;modify=none
```

The command above sets owner password to **`pdf123`**, and the user password to the one in the source document.

Note Passwords are not always retrievable. If you opened the document using the owner password, pdfleo is able to derive the user password. However, the opposite is not true. If the document is opened by user password, pdfleo can't derive the owner password. If you specify to copy the owner password, pdfleo will print a warning and replace it with the default owner password, `pdfleo`.

Copy Encryption Settings and Set Passwords and Permission

the following example sets owner password to pdf123, user password set to empty. Print permission to high resolution and modify option to none. both permissions require encryption to set to RC4-128 or AES-128.

```
C:\>pdfleo --encrypt=copy; --password-security=pdf123;;print=high;modify=none
```

Note that permission is based on the one in the source document if `--encrypt=copy` is specified. Otherwise it starts with full access.

3.3.4. Public key Recipients

This `--pubkey-recipient` switch specifies public key security options. Multiple switches can exist at the command line, with each specifying a recipient set. A recipient set comprises multiple recipients shared by the same permission flag. A recipient is identified by its X509 public key file.

```
C:\>pdfleo --encrypt=discard;AES-128 \
      --pubkey-recipient=thomas_tang.cer;joe_smith.cer;modify=none;print=low;extract=yes
```

The first part lists all certificate files, separated by semicolon. the second part specifies permission flags. The permission flag is based on source document if `copy` is specified at `--encrypt` switch, otherwise it starts with full access.

Hint: you can dump the list of recipients using `--info` switch after creating the document.

3.4. Extended Characters in Passwords

For maximum compatibility across applications, it is recommended to restrict password text to printable ASCII characters only. However if your existing PDF is encrypted with extended characters, or you plan to use extended characters in password, read this section. Extended characters are those not in the ASCII range (0x20~0x7e).

In PDF specification, password is interpreted as byte sequence. However, users often treat passwords text. Value of characters are subject to the interpretation of the code page. For example, Latin character Ä has value 0x80 under Mac Roman encoding, while on windows code page 1252 it is 0xC4. In order to uses to open encrypted files with the same password phrase across Windows and Mac, Acrobat converts the supplied password text into a special encoding native to PDF, called *PdfDocEncoding*. This encoding contains most characters in code page 1252 and MacRoman. Characters not in the encoding set are converted to spaces.

In order for users to have the same UI experience with Acrobat, pdfleo takes the same approach. Therefore you can encrypt PDFs with passwords containing Latin characters, and use the same password text in Adobe Reader. Note that other applications may take different approaches. Such applications may not be able to read PDFs encrypted in this manner.

The command below encrypts source PDF with owner password **`123456`**, and user password **`DésoléÄe`**.

```
C:\>pdfleo --encrypt=discard;AES-128;;no \
    --password-security=123456;DésoléÄe;print=none test3.pdf test8.pdf
```

3.5. Sample Usage

The following section demonstrates how you can use pdfleo to encrypt or decrypt PDF documents. Remember that encryption can be used in conjunction with other transforms, such as linearization and compression.

Preserving Encryption

Under preserve mode, the encryption settings are preserved in the target PDF. If the source PDF is not encrypted, the output is not encrypted either. If the source is encrypted, the output PDF preserves all encryption settings, such as encryption level, permission, passwords and recipient sets.

```
C:\>pdfleo --encrypt=preserve source.pdf target.pdf
```

`Preserve`is the default encryption mode and it can be omitted. The command line below has the same effect:

```
C:\>pdfleo source.pdf target.pdf
```

Stripping Encryption (Decryption)

By specifying discard without additional parameters, the encryption is removed from the PDF.

```
C:\>pdfleo --encrypt=discard source.pdf target.pdf
```

Encrypt with New Encryption Settings

If `discard` is followed by other parameters, the output PDF will be encrypted. When this is the case, security handler through `--password-security` or `--pubkey-security` is required. The `discard` mode sets encryption baseline to AES-128, 128 bits, clear text metadata and all access (permission). They can be modified by other command line parameters. The following statement encrypt the PDF with RC4-128, 80 bits, and metadata is encrypted. The following `--password-security` switch specifies the owner password as pdf128, user password empty; modify permission is set to `none`.

```
C:\>pdfleo --encrypt=discard;RC4-128;80;no \
  --password-security=pdf123;;modify=none \
  source.pdf target.pdf
```

You do not need to set all parameters explicitly. For example, the following command will encrypt document using the default encryption level - AES-128, clear text metadata.

```
C:\>pdfleo --encrypt=discard; \
  --password-security=pdf123;;modify=none \
  source.pdf target.pdf
```

The semicolon after `discard` can't be skipped. It indicates that the encryption parameters followed use the default value.

Encrypting with Settings Based on Source PDF

The copy mode uses encryption settings from the source PDF as the base line.

```
C:\>pdfleo --password=pdf123 --encrypt=copy --password-security=*;*;print=full
```

It is possible to change security setting under `copy` mode:

```
C:\>pdfleo --password=pdf123 --encrypt=copy;AES-128 \
  --password-security=*;*;print=full \
  source.pdf target.pdf
```

The command above changes encryption level to AES-128.

Encrypting with Public Key Security

By using `--pubkey-security` option, the output PDF can be encrypted with public key security.

```
C:\>pdfleo --encrypt=discard;AES-128 \
   --pubkey-security=john.smith.cer;jennifer.lopez.cer;modify=none;print=low \
   source.pdf target.pdf
```

The resulted document can only be opened by the persons who posses private keys of corresponding certificates: `john.smith.cer` and `jennifer.lopez.cer`. The permission is set to "no modification and low resolution printing".

Chapter 4. Technical Support

Morovia offers a wide variety of support services. To help you save time and money when you encounter a problem, we suggest you try to resolve the problem by following the options below in the order shown.

- Consult the documentation. The quickest answer to many questions can be found in the Morovia product documentation.
- Review the tutorial and sample applications. The tutorial steps you through the development process for a typical barcode application. The sample applications provide working code examples in several programming languages. All sample applications are extensively commented.
- Access Morovia Online. Morovia Online provides a knowledge base which documents the frequently asked questions and a web forum.

  The web address for knowledge base is http://support.morovia.com. You can ask question at support forum at http://forum.morovia.com.
- Contact Morovia Technical Support Service. The Technical Support service is provided for free up to 180 days after the purchase. Email Morovia support engineers at `support@morovia.com`.

Note If you purchased your software from our reseller, check to see if they provide support services before contacting Morovia.

Support services and policies are subject to change without notice.

Appendix A. Application Notes

A.1. PDF Date Format

PDF uses a special date format, which closely follows that of the international standard ASN.1 (Abstract Syntax Notation One), defined in ISO/IEC 8824. A date is a string of the form

```
D:YYYYMMDDHHmmSSOHH'mm'
```

where

- **YYYY** is the year
- **MM** is the month
- **DD** is the day (01–31)
- **HH** is the hour (00–23)
- **mm** is the minute (00–59)
- **SS** is the second (00–59)
- **O** is the relationship of local time to Universal Time (UT), denoted by one of the characters +, -, or Z (see below)
- **HH** followed by ' is the absolute value of the offset from UT in hours (00–23)
- **mm** followed by ' is the absolute value of the offset from UT in minutes (00–59)

The quotation mark character (') after HH and mm is part of the syntax. All fields after the year are optional. (The prefix D:, although also optional, is strongly recommended.) The default values for MM and DD are both 01; all other numerical fields default to zero values. A plus sign (+) as the value of the O field signifies that local time is later than UT, a minus sign (-) that local time is earlier than UT, and the letter Z that local time is equal to UT. If no UT information is specified, the relationship of the specified time to UT is considered to be unknown. Whether or not the time zone is known, the rest of the date should be specified in local time.

For example, December 23, 1998, at 7:52 PM, U.S. Pacific Standard Time, is represented by the string,

```
D:199812231952-08'00'
```

PDFLeo allows more relaxed syntax. The starting D: can be skipped. You can specify YYYY or YYYMMDD only. For example, the following examples are valid:

```
2010
D:2010
D:20101010
199812231952-08'00
```

A.2. Console Font in Windows

This program is fully Unicode compliant. In order to produce consistent output, all strings are outputted as UTF-8. If the output is redirected to a file, a UTF-8 capable viewer is required to view the contents correctly, such as notepad2, notepad++ and Internet Explorer under UTF-8 mode.

If you view the results at command line, set the console font to “Lucida Console” or a true type font that has most character sets available.

Appendix B. Create a Self-Signed Digital ID in Adobe Reader

If you are not using third-party digital ID, you can create your own self-signing digital ID. Many environments provide facilities to create digital ID, including Windows Certificate Manager, `CertUtil` and open source solution `openssl`. Here we explain how to create digital ID in Adobe Reader.

A digital ID file stores two pieces of information: an encrypted private key for signing and decrypting documents, and a public key containing a standard X509 certificate, which is used for validating signatures and encrypting documents. When creating digital ID files, we recommend that you stick with standard formats:

- PKCS#12 Digital ID File. Often the file has extension .pfx or .p12. It contains both private key and certificate. It is a binary file.
- Public Certificate File. Often the file has extension of .cer or x509. This file contains your public key and other information. The certificate file can be exported from Digital ID file.

Follow the steps below to generate a Digital ID file:

1. Select Document → Security Settings.
2. Select Digital IDs on the left, and then click Add ID button.
3. Specifiy A new digital ID I want to create now and click Next.
4. Choose New PKCS#12 Digital ID File to store it in a pfx file.
5. Type a name, email address, and other personal information for your digital ID. When you certify or sign a document, the name appears in the Signatures panel and in the signature field.
6. (Optional) To use Unicode values for extended characters, select Enable Unicode Support, and then specify Unicode values in the appropriate boxes.
7. Choose an option from the Key Algorithm menu. 2048-bit RSA offers more security than 1024-bit RSA, but 1024-bit RSA is more universally compatible.
8. From the Use Digital ID For menu, choose whether you want to use the digital ID for signatures, data encryption, or both. Click Next.
9. Specify a filename and location for the digital ID file.
10. Type a password; passwords are case-sensitive, must contain at least six characters, and may not contain double quotation marks or the following characters: ! @ # $ % ^ & * , | \ ; < > _. Type the same password in both the Password and Confirm Password boxes. Click Finish.
11. Export or send your certificate file to those who need to encrypt documents and sent to you.

Warning Backup a copy of your digital ID file. If your digital ID file is lost or corrupted, or if you forget your password, you cannot use that profile to open public key encrypted PDF documents.

Appendix C. Software License Agreement

IMPORTANT-READ CAREFULLY: This Morovia End-User License Agreement ("EULA") is a legal agreement between you (either an individual person or a single legal entity, who will be referred to in this EULA as "You") and Morovia Corporation (referred as "Morovia") for the Morovia software product that accompanies this EULA, including any associated media, printed materials and electronic documentation (the "Software Product"). The Software Product also includes any software updates, add-on components, web services and/or supplements that Morovia may provide to you or make available to You after the date You obtain Your initial copy of the Software Product to the extent that such items are not accompanied by a separate license agreement or terms of use. By installing, copying, downloading, accessing or otherwise using the Software Product, You agree to be bound by the terms of this EULA. If you do not agree to the terms of this EULA, do not install, access or use the Software Product; instead, you should return it to your place of purchase for a full refund.

The Software Product is protected by intellectual property laws and treaties.

1. GRANT OF LICENSE. This Section of the EULA describes Your general rights to install and use the Software Product The license rights described in this Section are subject to all other terms and conditions of this EULA.

General License Grant to Install and Use Software Product. You may install and use one copy of the Software Product on a single computer, device, workstation, terminal, or other digital electronic or analog device ("Device"). You may make a second copy of the Software Product and install it on a portable Device for the exclusive use of the person who is the primary user of the first copy of the Software Product. A license for the Software Product may not be shared.

Alternative License Grant for Storage/Network Use. As an alternative to the rights granted in the previous section, You may install a copy of the Software Product on one storage Device, such as a network server, and allow individuals within Your business or enterprise to access and use the Software Product from other Devices over a private network, provided that You acquire and dedicate a license for the storage Device upon which the Software Product is installed and each separate Device from which the Software Product is accessed and used. A license for the Software Product may not be used concurrently on different Devices.

2. DESCRIPTION OF OTHER RIGHTS AND LIMITATIONS.

Copy Protection. The Software Product may include copy protection technology to prevent the unauthorized copying of the Software Product or may require original media for use of the Software Product on the Device. It is illegal to make unauthorized copies of the Software Product or to circumvent any copy protection technology included in the Software Product.

Limitations on Reverse Engineering, Decompilation, and Disassembly. You may not reverse engineer, decompile, or disassemble the Software Product, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation.

Separation of Component Parts. The Software Product is licensed as a single product. Its component parts may not be separated for use on more than one Device unless expressly permitted by this EULA.

No rental, leasing or commercial hosting. You may not rent, lease, lend or provide commercial hosting services to third parties with the Software Product.

Software Transfer. Except as specified in this Section, the initial licensee of the Software Product may make a one-time permanent transfer of this EULA and Software Product only directly to an end user. This transfer must include all of the Software Product (including all component parts, the media and printed materials, any upgrades, this EULA, and, if applicable, the Certificate of Authenticity). Such transfer may not be by way of consignment or any other indirect transfer. The transferee of such one-time transfer must agree to comply with the terms of this EULA, including the obligation not to further transfer this EULA and Software Product.

Termination. Without prejudice to any other rights, Morovia may terminate this EULA if You fail to comply with the terms and conditions of this EULA. In such event, You must destroy all copies of the Software Product and all of its component parts.

For Trial Software. This is not free software. You are hereby licensed to use this software for evaluation purposes without charge for limited functions. If you want use this software with full functions a registration fee is required. You must completely remove the software from your computer upon the expiration of the trial period.

3. INTELLECTUAL PROPERTY RIGHTS. All title and intellectual property rights in and to the Software Product (including but not limited to any images, photographs, animations, video, audio, music, text, and "applets" incorporated into the Software Product), the accompanying printed materials, and any copies of the Software Product are owned by Morovia or its suppliers. All title and intellectual property rights in and to the content that is not contained in the Software Product, but may be accessed through use of the Software Product, is the property of the respective content owners and may be protected by applicable copyright or other intellectual property laws and treaties. This EULA grants you no rights to use such content. If this Software Product contains documentation that is provided only in electronic form, you may print one copy of such electronic documentation. You may not copy the printed materials accompanying the Software Product.

4. BACKUP COPY. After installation of one copy of the Software Product pursuant to this EULA, you may keep the original media on which the Software Product was provided by Morovia solely for backup or archival purposes. If the original media is required to use the Software Product on the Device, you may make one copy of the Software Product solely for backup or archival purposes. Except as expressly provided in this EULA, you may not otherwise make copies of the Software Product or the printed materials accompanying the Software Product.

5. APPLICABLE LAW. If you acquired this Software Product in Canada, unless expressly prohibited by local law, this EULA is governed by the laws in force in the Province of Ontario, Canada; and, in respect of any dispute which may arise here under; you consent to the jurisdiction of the federal and provincial courts sitting in Toronto, Ontario. If this Software Product was acquired outside Canada, then local law may apply.

6. No LIABILITY FOR DAMAGES. In no event shall the author of this Software be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or any other pecuniary loss) arising out of the use of or inability to use this product, even if the Author of this Software has been advised of the possibility of such damages.

Because some states' jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitation may not apply to you. In this case the Author of this Software will only be liable for the amount of money you spent for this SOFTWARE PRODUCT in exchange for the return of the product, all copies, registration papers and manuals, and all materials that constitute a transfer of ownership from the customer back to the Software Author.

Index