



AFP2web Version 3.x

Benutzerhandbuch und Referenz

Maas High Tech Software GmbH

Integrate the future today!



AFP2web ist ein eingetragenes Warenzeichen der Maas High Tech Software GmbH.

Warenzeichen oder eingetragene Warenzeichen anderer Hersteller, die in dieser Publikation Verwendung finden: AFP (IBM Corporation); Helvetica, Times (Linotype Hell AG); ITC Zapf Dingbats (International Typeface Corporation); PostScript, Acrobat (Adobe Systems Incorporated); Windows (Microsoft Corporation). Alle in dieser Publikation verwendeten Soft- und Hardware-Bezeichnungen, sowie Markennamen und Produktbezeichnungen der jeweiligen Firmen unterliegen im allgemeinen warenzeichen-, marken- oder patentrechtlichem Schutz

Copyright © 2008 Maas High Tech Software GmbH

Alle Rechte vorbehalten.

Maas High Tech Software GmbH,

Hornbergstr. 49, D-70794 Filderstadt, Deutschland

Telefon: +49 (0) 711 – 77917 – 0

Internet: <http://www.maas.de>

Inhaltsverzeichnis

Vorwort 7

 AFP2web und die AFP2web Scripting Facility..... 9

 Neues seit AFP2web Version 2.x 13

 Merkmale, die nicht länger unterstützt werden 17

 Wichtige Hinweise 19

 Von AFP2web Version 2.x nach Version 3.x migrieren..... 21

Kapitel 1: Einführung in AFP2web 23

 1.1 Überblick der AFP2web Funktionalität 25

 Unterstützte Formate 26

 AFP2web Kernkomponenten 27

 Methoden zur Integration von AFP2web 28

 1.2 AFP2web Anwendungsszenarios 29

 Dokument-Archivierung und Index-Aktualisierung 30

 On-the-fly Konvertierung für PDF On-Demand 31

 AFP Viewer für Workstations 32

 Dokument-Versand per Post (Print), E-Mail, oder Fax 33

Kapitel 2: AFP2web Benutzerhandbuch 35

 2.1 AFP2web installieren 37

 Systemvoraussetzungen 37

 Installation unter Windows 39

 Installation unter Unix 39

 Das Installationsergebnis 40

 AFP2web deinstallieren 41

 2.2 AFP2web konfigurieren 43

 Parameter in der INI-Datei festlegen 44

 Musterdokumente konvertieren und die Konfiguration anpassen 44

 AFP Ressourcen festlegen 45

 2.3 AFP2web anwenden 49

Der Schnelleinstieg	49
AFP2web in der Kommandozeile starten, stoppen	50
Die Konvertierung steuern	51
Dokumente für die Konvertierung auswählen	52
Dokumentseiten für die Konvertierung auswählen	53
AFP Indexelemente verarbeiten	54
Ausgabedokumente in Unterverzeichnisse ausgeben	56
Die Verarbeitung von eingebetteten Objekten (Included Objects)	57
Hinweise zur Barcode Unterstützung	59
PDF Lesezeichen für Indexelemente anlegen	61
Einen Standardtext als Wasserzeichen unterlegen	62
Konvertierung abbrechen, wenn eine Ressource fehlt	63
Die LOG-Datei nutzen	64

Kapitel 3: Die Behandlung von Schrift bei der Konvertierung 67

3.1 Die Konfigurationsdateien und Voreinstellungen für Schrift	69
Die INI-Datei (afp2web.ini)	70
Die Fontmapping Datei (mapping.def)	71
Code Page Dateien	72
Standard Schriftzuordnung	73
3.2 Die Verarbeitungsmodi für Schrift in AFP2web	75
Das Standardverhalten von AFP2web	75
Schriftersetzung und -referenzierung	79
Schriftersetzung und -einbettung	86

Kapitel 4: AFP2web Scripting Facility Benutzerhandbuch 89

4.1 Grundlagen der AFP2web Scripting Facility	91
Scripting Facility Anwendungen	91
Scripting Facility Interaktion mit AFP2web	94
Scripting Facility Besonderheiten	99
4.2 AFP2web Scripting Facility verwenden	103
Aktivierung der Scripting Facility	104
Einstellungen für die Perl Umgebung	106
Weitere Verwendungshinweise	108

Kapitel 5: AFP2web Referenz 109

5.1 Gegenüberstellung: INI-Parameter, Kommandozeilenoptionen	111
5.2 Parameter der afp2web.ini	119
Abschnitt [Settings]	120

Parameter des Abschnitts [FORMDEF]	141
Parameter des Abschnitts [AFPColorTable]	142
Frei definierbare Abschnitte für Logische Schnittstellen	143
5.3 AFP2web Kommandozeilenoptionen	145
5.4 Parameter der mapping.def	163
[CHARSET RENDERING] Abschnitt	166
[XFONT] Abschnitt	169
[CHARSET] Abschnitt	170
[FGID] Abschnitt	171
[FONTSUFFIX] Abschnitt	172
[PDFFONT] Abschnitt	173
[WINFONT] Abschnitt	174
[UNIXFONT] Abschnitt	175
[AFPFONT] Abschnitt	176
[CODEPG] Abschnitt	177
5.5 Code Page Dateien (CP-Dateien)	179
5.6 Die Ausgabe von Schriftinformationen in der LOG-Datei	181
Das Format der LOG-Meldungen zur Schrift	181
Verwendete Kürzel in den LOG-Meldungen zur Schrift	182
Beispiele für LOG-Meldungen zur Schrift	186
5.7 Schriftzuordnung laut IBM Namenskonvention für Ressourcen	191
Die AFP Schriftnamenskonvention	192
5.8 Scripting Facility Schnell-Referenz	197
Skriptrouinen und Methoden	198
Methoden und deren Verfügbarkeit	205
5.9 Scripting Facility API Reference	213
Overview	213
afp2web.pm	214
a2w::Config (mhtml [DEPRECATED])	220
a2w::Document (mhtmlDocument [DEPRECATED])	223
a2w::Font (mhtmlAFPFont [DEPRECATED])	231
a2w::Index (mhtmlAttributeElement, mhtmlIndexElement, mhtmlPagePageGroupIndex [DEPRECATED])	242
a2w::Kernel [NEW]	248
a2w::Line [NEW]	250
a2w::MediumMap [NEW]	257
a2w::NOP (mhtmlNOP [DEPRECATED])	260
a2w::Overlay (mhtmlResOverlay [DEPRECATED])	263
a2w::PSEG [NEW]	269
a2w::Page (mhtmlPage [DEPRECATED], mhtmlPage [DEPRECATED]) ..	271

a2w::Text (mhtTextObject [DEPRECATED])	291
5.10 AFP2web Meldungen	299
AFP2web Fehlermeldungen	300
MHTIMG Library Meldungen	313
Kapitel A: Anhang	319
A.1 Migration der Scripting Facility Module zu Version 3.x	321
A.2 Scripting Facility Beispiele	325
Einführung	325
dumpIniParms.pm: INI Parameters ausgeben	331
autosplit.pm: Eine AFP Spooldatei in Dokumente und Seiten trennen	335
dumpPageTextObjs.pm: Text und Schriftinformationen ausgeben	345
sortPageTextObjs.pm: Textobjekte nach der Print-Reihenfolge sortieren	352
dumpMediumMaps.pm: Liste der Medium Maps ausgeben	358
dumpNOPs.pm: Inhalte von NOP Feldern ausgeben	362
eyecatcher.pm: Text Finden und Extrahieren	366
addBookmarks.pm: PDF Bookmarks (Lesezeichen) erzeugen	376
addWatermark.pm: Eine Hintergrundgrafik auf der Seite einfügen	380
addAnnotations.pm: Eine Hyperlink-Verknüpfung hinzufügen	382
afp2xml.pm: Skript zur Umsetzung von AFP zu XML	385
A.3 Die PDF/A Unterstützung in AFP2web	391
Zur Unterstützung von PDF/A in AFP2web	391
Zum Inhalt	391
Wichtige Hinweise	391
INI Parameter und Kommandozeilenoptionen	392
Schriften	393
Metadaten im XMP Format	394
Merkmale, die für PDF/A nicht zu verwenden sind	395
A.4 Frequently Asked Questions (FAQs)	397
Stichwortverzeichnis	399

Vorwort

Dieses Vorwort enthält

- eine Kurzeinführung in das AFP2web Benutzerhandbuch
- eine Auflistung der neuen Merkmale von AFP2web Version 3.x
- wichtige Hinweise für die Verwendung von AFP2web
- Hinweise für die Migration von AFP2web 2.x nach Version 3.x

AFP2web und die AFP2web Scripting Facility

AFP2web ist ein Werkzeug für die intelligente Dokumentenerkennung, -trennung, -konvertierung und -verteilung von AFP Massendruckdaten.

AFP2web konvertiert AFP Druckdaten in die Standardformate: PDF, PNG, TIFF, ASCII und XML und PDF Dokumente zu AFP. Mit AFP2web werden AFP Daten für das Web verfügbar und dennoch bleibt die Leistung eines sicheren und mächtigen IBM S/390 oder Unix Systems erhalten. AFP2web dient der Aufbereitung von Dokumenten für das Web, für die Archivierung, Indizierung und für den Austausch in dokumentorientierten Workflows. AFP2web bietet ein qualitativ hochwertiges originalgetreues Konvertierungsergebnis und ist alleine oder auch als Einzelkomponente in komplexen Anwendungsszenarios einsetzbar.

Die AFP2web Scripting Facility ist eine Erweiterung zu AFP2web. Sie bietet eine Scripting Schnittstelle, mit der man intelligent Dokumente erkennen, trennen, indizieren und weiterverarbeiten kann.

Wer sollte dieses Dokument lesen

Dieses Dokument richtet sich an alle Anwender von AFP2web. Für den Einsatz von AFP2web alleine genügt es, eine Job-Umgebung einzurichten und die Konfigurationsdateien von AFP2web anzupassen. Anwender, die AFP2web anpassen möchten, finden Informationen wie dieses mit der Scripting Facility sich realisieren lässt.

Zum Inhalt

Dieses Dokument ist in folgende Kapitel aufgeteilt:

- Dieses Vorwort gibt einen Überblick über die neuen Merkmale von AFP2web Version 3.x, wichtige Hinweise und Empfehlungen für die Migration von der Version 2.x zur Version 3.x.
- Eine Einführung in AFP2web, dessen Funktionsumfang und mögliche Anwendungsszenarios.
- Das AFP2web Benutzerhandbuch mit der Beschreibung der Installation, Konfiguration, des Programmaufrufs und der Verwendung einfacher Verarbeitungsfunktionen.
- Eine allgemeine Einführung in die Behandlung von Schrift durch AFP2web. Dieses Kapitel hebt hervor, dass AFP2web standardmäßig die AFP Originalschriften

verwendet, beschreibt auch weitere Verarbeitungsmodi für die Verwendung von Ersatzschriften.

- Das AFP2web Scripting Facility Benutzerhandbuch dient als Einführung zur Scripting Facility.
- Die AFP2web Referenz enthält die detaillierte Beschreibung der Syntax und Funktion der AFP2web Konfigurationsparameter (die INI-Datei, Optionen für die Kommandozeile, mapping.def), die AFP Namenskonvention für die Zuordnung zu Ersatzschriften, die Scripting Facility Schnellreferenz, die englischsprachige Scripting Facility API Referenz und schließlich die Liste der Codes und Fehlermeldungen.
- Der Anhang bietet eine Checkliste für die Migration von Scripting Facility Modulen zu der Version 3.x und eine Einführung in die Scripting Facility Beispiele.

Weitere Informationen

Weitergehende Informationen zu AFP finden Sie unter <http://www.printers.ibm.com>.

Informationen zu Perl finden Sie auf der Perl Homepage <http://www.cpan.org/>.

Kontakt

Dieses Dokument beinhaltet die Informationen, die Sie für den Einstieg und die Verwendung von AFP2web und der Scripting Facility benötigen. Auf Anfrage erstellt Maas High Tech Software GmbH für Ihre Druckdaten gern ein komplettes Scripting Facility Modul. Sie können dieses Modul als Ausgangspunkt für die weiteren Anpassungen verwenden. Wir empfehlen ebenso die Verwendung der Scripting Facility Beispiele.

Maas High Tech Software GmbH bietet professionelle Beratung für die Entwicklung und den Einsatz von AFP2web-basierten Lösungen.

Weitere Informationen finden Sie im Web unter der Adresse

<http://afp2web.de>

oder per E-Mail:

E-mail: afp2web@maas.de

Konventionen

XML Daten

XML Dokumente sind für die bessere Lesbarkeit mit Einrückungen und Zeilenumbrüchen dargestellt. Die Codierungsbeispiele und Systemausgaben verwenden eine Schrift mit einheitlicher Schriftbreite.

Darstellung von XML Daten

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE Index SYSTEM "Index.dtd" >
<Index Doctype="PDF" DocName="pdf/xml/INSURE_s3k.0.pdf"
PageCount="3" Size="16726">
  <PageGroupIndex PageGroup="00000001">
    <Data>
      <Name>Insured</Name>
      <Value>Geoffrey R Stephens</Value>
    </Data>
  </Data />
</PageGroupIndex>
</Index>
```


Neues seit AFP2web Version 2.x

AFP2web ist ein Produkt, das Sie in jeder Anwendung beliebiger Komplexität verwenden können. AFP2web bietet grundlegende Funktionen an, die mit Hilfe der Scripting Facility angepasst werden kann.

Dank der besonderen Leistungsfähigkeit bei der Umsetzung von Schrift, PDF Sicherheitseinstellungen und der mächtigen Scripting Facility hat sich die AFP2web Version 3.x von einem reinen Konvertierungswerkzeug zu einer Basis-Technologie für die Dokumentkonvertierung und -verarbeitung entwickelt.

Dieser Abschnitt fasst die neuen Produktmerkmale von AFP2web Version 3.x seit der Version 2.x zusammen.

Die Konvertierung von PDF zu AFP

AFP2web konvertiert PDF Dokumente in ein AFP Format, das dem Standard MOD:CA P oder MOD:CA Interchange Set 2 entspricht.

Neue INI Parameter sind eingeführt: PrintableLineWidth, CurveSamplingFactor, AFPComplianceLevel. Der INI Parameter PageRotation ist erweitert.

Der mapping.def Abschnitt [CHARSET RENDERING] ermöglicht eine feinere Kontrolle bei der Rasterung von Schrift. Neu ist der Abschnitt [AFPFONT] für die Zuordnung zu Ersatzschriften.

Ist Schriftrasterung für die Konvertierung von PDF zu AFP aktiviert, werden die eingebetteten Schriften ohne Änderung verwendet. AFP2web sucht auch nach externen, referenzierten Schriften.

Erweiterungen in der Unterstützung von Schrift

AFP2web bietet Verbesserungen in der Unterstützung von Schrift:

- Font Rasterizing ist ein neuer Verarbeitungsmodus für Schrift, der die ursprünglichen AFP Schriften benützt und zu einer 100-prozentigen Originaltreue im Konvertierungsergebnis führt.
- Font Referencing/Substitution und Font Embedding sind weitere Verarbeitungsmodi für Schriften. Die AFP Schriften werden durch zugeordnete Ersatzschriften ersetzt. Die unterstützten Formate sind: Type 1, TrueType und OpenType mit TrueType als einziges Subset.

Neues für AFP als Eingabeformat

AFP2web bietet Verbesserungen in der Verarbeitung von AFP Spooldateien:

- Unterstützung von farbigen Grafiken als IOCA FS 42 und 45, GOCA und Object Containers (TIFF, JPEG,...).
- AFP2web unterstützt Barcodes: Code 128. AFP2web verwendet hier den Character Set B von Code 128, um die Barcodes zu zeichnen, da dieser Zeichensatz Kleinbuchstaben, Großbuchstaben, Zahlen und Satzzeichen enthält.
- AFP2web unterstützt auch spezifische Formate des zweidimensionalen DataMatrix 2D Barcodes. Nähere Informationen hierzu finden Sie unter ["Hinweise zur Barcode Unterstützung" auf Seite 59.](#)
- Die erweiterte Unterstützung der Kompressionsalgorithmen für Grafiken: Uncompressed, IBM RL4, LZW, G3, G4, Old JPEG, JPEG, IBM MMR, IOCA (FS 10, 11, 42, 45), CCITTRLE, CCITTRLEW, IT8CTPAD, IT8LW, IT8MP, IT8BL, PIXARFILM, PIXARLOG, DEFLATE, ADOBE_DEFLATE, THUNDERSCAN, DCS, NEXT, SGILOG, SGILOG24. (Hinweis: AFP2web unterstützt nicht CMYK (32Bits/Pixel) JPEGs.)
- FTP Zugang zu entfernten Ressourcen. AFP2web kann für den Zugriff auf Ressourcen über FTP-Server konfiguriert werden.

Neues für die Ausgabe nach PDF und PDF/A

AFP2web bietet Verbesserungen in der PDF Ausgabe:

- Unterstützt 40/128 Bit bei den PDF Sicherheitseinstellungen, somit bleiben die Kontrolle und die dazugehörigen Rechte über das Dokument beim Ersteller.
- Es besteht die Möglichkeit, für die erzeugten PDF Dateien Adobe Acrobat Reader Einstellungen zu definieren.
- Die Möglichkeit der Volltextsuche in PDF Dokumenten, auch wenn diese die originalen AFP Schriften verwenden.
- Reduzierung der PDF Dateigröße durch die Auswahl des bestmöglichen Kompressionsverfahrens für Schwarz/Weiß oder farbige Grafiken.
- Erweiterte Möglichkeiten zur Anpassung von Dokumenten: Änderung von Text, Indexdaten, Kommentaren, Lesezeichen und vieles mehr.

AFP2web unterstützt das neue Ausgabeformat PDF/A, das dem Standard der Stufe PDF/A-1b entspricht. Der Standard wird definiert in ISO 19005-1. Document management — Electronic document file format for long-term preservation — Part 1: Use of PDF 1.4 (PDF/A-1) Reference number ISO 19005-1:2005(E)

PDF/A-konforme Dokumente sind selbstdefinierende PDF Dateien, die für die Langzeit-Archivierung verwendet werden können. Der Standard basiert auf PDF Version 1.4 mit weiteren Einschränkungen in den PDF Funktionen.

Merkmale für Rasterformate

AFP2web bietet bedeutende Verbesserungen in der Unterstützung von Rasterformaten wie TIFF an:

- Für die Verwendung von Rasterformaten als Eingabe, wurden die unterstützten Kompressionsalgorithmen Uncompressed, LZW, G3, G4, Old JPEG und JPEG verbessert.
- Für die Ausgabe in ein Rasterformat bietet es erweiterte Möglichkeiten zur Anpassung von Dokumenten: Änderung von Text, Indexdaten, Kommentaren, Lesezeichen und vieles mehr.

Für die Ausgabe von TIFF werden die folgenden AFP2web-spezifischen Tags hinzugefügt, um die Meta-Informationen aus der INI-Datei einzubetten.

<i>TIFF Tag</i>	<i>Inhalt</i>
667	Keyword
668	Subject
669	Title

Hinweis: Diese Tags können in einigen TIFF Viewern beim Öffnen von AFP2web erzeugten TIFF-Dateien zu einer Warnung "Unknown field/tag" führen, der aber keine weitere Bedeutung beizumessen ist.

Neue Merkmale der Scripting Facility

Die Scripting Facility ermöglicht den Zugriff auf den druckbaren und nicht-druckbaren Inhalt einer Dokumentseite noch vor der Konvertierung. Mit den neuen Funktionen können Sie Dokumentinhalte hinzufügen, ändern oder löschen.

Die Eingabe und Ausgabe kann direkt in den Speicher gelenkt werden, damit zeitaufwändiges Lesen und Schreiben in Dateien vermieden wird.

Sie finden weitere Informationen über die Möglichkeiten der Scripting Facility in diesem Benutzerhandbuch. AFP2web bietet als Beispiele einige Scriptmodule, die Sie als Ausgangspunkt für Ihre eigenen Lösungen verwenden können.

Betriebssystemunterstützung

Die Liste der unterstützten Betriebssysteme ist erweitert. Siehe hierzu "Systemvoraussetzungen" auf Seite 37.

Neue INI-Parameter und Optionen für die Kommandozeile

Neu sind die folgenden INI-Parameter und Optionen für die Kommandozeile:

- Der Abschnitt [AFPCOLORTable] in der INI-Datei ordnet jede Farbe der AFP Standard OCA Farbtabelle einem RGB-Wert zu. Die entsprechende Option für die Kommandozeile ist:
-clr:<colorname>,<r>,<g>,,<colorname>,...
- Autosplit ist ein INI-Parameter zur automatischen Trennung von AFP Spools in Dokumente und Dokumentseiten und beeinflusst die Ausführung der Scripting Facility.
- CMYKTORGB dient der Konvertierung von IOCA FS45 CMYK Farbebenen zu RGB. Die entsprechende Option für die Kommandozeile schaltet diese Funktion aus: -nocmyktorgb.
- FilenamePattern definiert das Muster für die Namen der Ausgabedateien.
- LoggingLevel ist erweitert um: 8, SF, RES, OLY, PSEG, FNTR, CDP, MMAP, FDEF, IOB. Die entsprechende Option für die Kommandozeile ist -ll.
- PDFSecurity definiert PDF Sicherheitseinstellungen.
- PDFUIOptions, PDFWinOptions definiert Einstellungen für die Darstellung von Dokumenten im PDF Viewer. Die entsprechenden Optionen für die Kommandozeile sind -pui und -pwin.
- SkipPage, SkipObjectSize sind neue Parameter und legen fest, welche Seiten in einem AFP Spool ignoriert werden können.
- Benutzerdefinierte Abschnitte in der INI-Datei zur Definition von FTP Servern, die den Zugriff auf entfernte AFP Ressourcen ermöglichen.

Weitere Merkmale

Weitere neue Merkmale von AFP2web sind:

- Behandlung von Grafiken: Schnellere Verarbeitung, bessere Komprimierung, kleinere PDF Dateien.
- Farbige Grafiken werden für die Ausgabe in Schwarz/Weiß zu Graustufenbildern konvertiert.
- AFP2web zeigt die Versionsnummer im Format: AFP2web Version <Versionsnummer> [Built for <Name des Betriebssystems> on <Datum> at <Zeit>].

Merkmale, die nicht länger unterstützt werden

Dieser Abschnitt nennt die Produktmerkmale von AFP2web Version 3.x die nicht länger unterstützt werden.

Maas Font Metriken

AFP2web verwendet die Originalschriften und benötigt keine weitere Informationsquellen zur Formatierung einer Schrift.

Fontmetriken werden daher als Leistungsmerkmal in AFP2web nicht länger unterstützt.

Wichtige Hinweise

Bitte setzen Sie AFP2web und die AFP2web Scripting Facility mit der erforderlichen Sorgfalt ein. Unsachgemäße Einstellungen der Parameter von AFP2web können das Konvertierungsergebnis negativ beeinflussen. Im schlimmsten Falle werden Dokumente für die Archivierung unbrauchbar.

Wir empfehlen daher:

Gründlich testen!

Testen Sie für jeden Dokumententyp die Einstellungen, die Sie in AFP2web vornehmen, gründlich. Hierzu gehören die Konfigurationsparameter (INI-Datei, Kommandozeilenoptionen), die Ressourcen und die Einstellungen für Schriften. Größte Sorgfalt ist vor allem bei der Aufbereitung der Dokumente für die Archivierung geboten.

Nur Sie kennen Ihre Schriften!

AFP2web kann nicht automatisch kundenspezifische Anpassungen in Ihren Ressourcen (wie Code Pages) erkennen. Kontaktieren Sie im Zweifelsfall die Maas High Tech Software GmbH.

Dokumente sichern!

AFP2web Produkte und Optionen sind nicht für die Datensicherung eingerichtet. Es liegt in Ihrer Verantwortung, die Daten, die für die Verarbeitung durch AFP2web bereitgestellt werden und die aus AFP2web entstehen, zu sichern.

Sichern Sie Ihre letzte AFP2web Installation!

Sie sollten die alten AFP2web Konfigurationsdateien unbedingt sichern, bevor Sie eine neue Version von AFP2web installieren. Sie können Ihre alte Anpassungen gemäß den neuen Anforderungen von AFP2web abändern.

Wir empfehlen, dass Sie die alten Versionen der folgenden Dateien sichern:

- INI-Datei
- afp2web.pm (bzw. Ihre Skriptmodule für die Scripting Facility)
- mapping.def
- *cp-Dateien
- die zu verwendende Ersatzschriften für PDF (*.pfm und *.pfb Dateien)

Tipp: Vor der Installation benennen Sie das Zielverzeichnis Ihrer vorigen Version von AFP2web um (Beispiel: A2W_version_datum). Erstellen Sie ein neues Zielverzeichnis mit dem alten Namen für die neue Installation (Beispiel: A2W). Nach der Installation können Sie bei Bedarf auf die Sicherungskopien zurückgreifen und die bereits vorgenommenen Anpassungen in die neuen Konfigurationsdateien integrieren.

Verwenden Sie die AFP2web Scripting Facility mit Sorgfalt!

Der Einsatz von AFP2web ohne die nötige Sorgfalt kann zu einem Datenverlust führen. Möglich ist die Beeinträchtigung des Erscheinungsbilds z.B. durch falsche Schriftzuordnung oder der Verlust der Dokumentidentifikation infolge inkorrektur Dokumententrennung.

Wir empfehlen daher:

- Verwenden Sie für unterschiedliche Arten von Spooldateien auch verschiedene Instanzen des Perl Skripts `afp2web.pm`. Sie können das Standard Skript ersetzen oder mit Hilfe der Kommandozeilenoption `-sp` das jeweilige Skript verwenden.
- Testen Sie Ihre Scripting Facility Module. Ihre Skripten steuern das Überspringen von Seiten, die Dokumententrennung oder die Behandlung von Indexdaten.

Von AFP2web Version 2.x nach Version 3.x migrieren

AFP2web Version 3.x bietet neue Funktionen, für die bestimmte Anpassungen in den Konfigurationsdateien und in den Skripting-Modulen erforderlich sind.

Wichtig: Jede Anpassung in den vorherigen AFP2web Versionen sind heute entweder nicht mehr erforderlich oder müssen gemäß den neuen nachfolgend beschriebenen Anforderungen verändert werden.

Änderungen in der mapping.def

AFP2web bietet eine verbesserte Unterstützung von Schriften an. AFP2web verwendet die AFP Schrift, wenn diese als Ressource zugänglich ist. Somit ist die Festlegung von Ersatzschriften, wie sie in der Version 2.x noch nötig war, heute nicht länger erforderlich. Sie können daher in den meisten Fällen auf Anpassungen in der mapping.def verzichten.

Für die wenigen Fälle, in denen Sie eine Ersatzschrift benötigen, müssen jedoch Einträge in der mapping.def vorhanden sein.

AFP2web Version 3.x kennt die folgenden neuen Definitionen in der mapping.def:

- Im [PDFFont] Abschnitt werden Präfixe vor Schriftnamen nicht mehr verwendet.
- [CHARSET RENDERING] ist ein neuer Abschnitt und legt fest, zu welchem Zeitpunkt die vorgegebene AFP2web Schriftverarbeitung überschrieben wird.
- [FONTSUFFIX] ist ein neuer Abschnitt und legt eine Namenskonvention (die Suffixe) für Schriftdateien fest. Da es in dieser Hinsicht keinen gültigen Standard gibt, bietet dieser Abschnitt die Möglichkeit, mehrere Suffixe frei zu bestimmen.

Änderungen in den CP-Dateien

AFP2web Version 3.x hat das Format der Code-Page-Dateien (CP-Dateien) für Unicode-Unterstützung erweitert. Diese Erweiterung ermöglicht die Volltext-Suche in PDF-Dateien.

Schrift-Ressourcen definieren und zur Verfügung stellen

In AFP2web ist es nicht länger erforderlich, Ersatzschriften für AFP Schriften zu verwenden. Ist die AFP Schrift als Ressource inline oder extern verfügbar, so verwendet AFP2web diese Schrift.

Für den Fall, dass Sie eine Ersatzschrift verwenden wollen, können Sie weiterhin die Schriftersetzung durch AFP2web nutzen. AFP2web unterstützt einfache Schriftersetzungsfunktionalitäten. Die AFP Schrift-Ressource liefert die nötigen Informationen für die Bestimmung einer Ersatzschrift. Zusätzlich lassen sich in der `mapping.def` Namen und Suffixe für die Dateinamen festlegen.

Hinweis: In der früheren Version von AFP2web, musste eine Schrift für die Konvertierung nach TIFF im System installiert sein. AFP2web Version 3.x wird stattdessen eine erforderliche Schrift zur Laufzeit installieren und am Prozessende wieder deinstallieren.

Detaillierte Information hierzu finden Sie in *"Die Behandlung von Schrift bei der Konvertierung"* auf Seite 67.

Scripting Facility Änderungen

Sie müssen Ihre selbst-erstellte Skriptmodule für die Scripting Facility anpassen.

Geändert sind die Namen der Packages und Methoden. Eine Liste der Änderungen finden Sie in *"Migration der Scripting Facility Module zu Version 3.x"* auf Seite 321.

Die Scripting Facility bietet noch mehr Funktionalität. Sie finden einen Überblick in *"Scripting Facility Schnell-Referenz"* auf Seite 197.

Die detaillierte API Referenz finden Sie in diesem Benutzerhandbuch unter *"Scripting Facility API Reference"* auf Seite 213.

Kapitel 1: Einführung in AFP2web

Diese Einführung beschreibt:

- die für die Konvertierung unterstützten Eingabe- und Ausgabeformate
- AFP2web Kernkomponenten
- Komponenten zur Erstellung einer AFP2web Anwendung
- typische AFP2web Anwendungsszenarios

1.1 Überblick der AFP2web Funktionalität

In der Vergangenheit galt proprietäre AFP Technologie als Standard für Massendruckdaten. Heute öffnet AFP2web die Tür zu einer effektiveren Kommunikation. Mit Hilfe von AFP2web und durch die Umwandlung zu den De-facto-Internetformaten PDF, JPEG, ASCII, TIFF oder XML werden AFP Dokumente für viele Systeme verfügbar. AFP2web ermöglicht darüber hinaus auch eine intelligente Dokumentenerkennung, -trennung, -konvertierung und -verteilung von AFP Massendruckdaten.

Ein Hauptaugenmerk ist die Ausgabe von XML für die einfache Integration mit zukünftigen Internet-basierenden Drucklösungen und für Data-Mining Lösungen zur Erhebung von Daten aus Druckdaten. Damit hat AFP2web seine Grenzen noch nicht ausgeschöpft, inspiriert vielmehr zu neuen Anwendungsmöglichkeiten. Weitere Informationen über AFP2web Lösungen finden Sie im Web unter <http://afp2web.com>.

Im Folgenden beschreiben wir:







- Die unterstützten Formate
- AFP2web Kernkomponenten
- Methoden zur Integration von AFP2web
- AFP2web Anwendungsszenarios

Unterstützte Formate

Es gibt eine Lösung für jede Konvertierungsanforderung. AFP2web bietet eine Basisfunktionalität, die durch weitere Optionen ergänzt werden kann. Fast jede Kombination von Ein- und Ausgabeformaten ist möglich, wie z.B. AFP oder TIFF nach ASCII, JPEG, PDF, TIFF, XML und andere Formate.

Die folgende Matrix zeigt die verfügbaren Formatoptionen.

Eingabe- und Ausgabeformate von AFP2web

Output Input	AFP	TIFF	PDF	PDF/A	JPEG	PNG	ASCII	XML
	X	X	X	X	X	X	X	X
 1	X	X	X	X	X	X		
	X	X	X	X ²	X	X	X	X
	X	X	X	X	X	X	X	X
	X	X	X	X	X	X	X	X
	X	X	X	X	X	X	X	X

1 = TIFF, PCX, BMP, GIF, JPEG

2 = planned / geplant

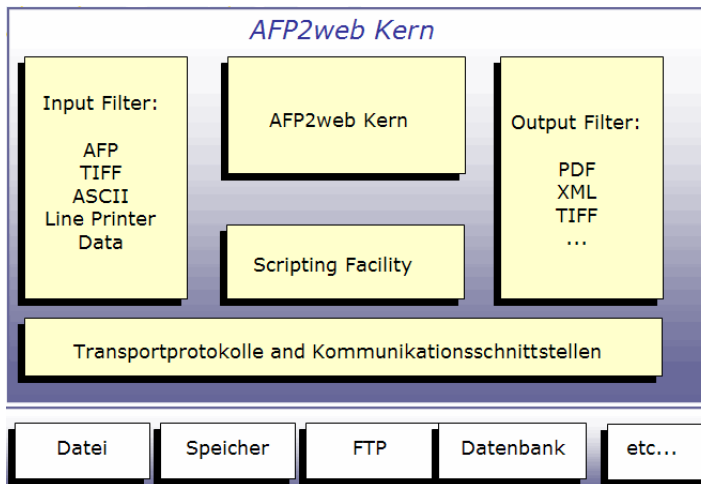
AFP2web Kernkomponenten

AFP2web ist auf die Zukunft ausgerichtet. Die interne Struktur dieses einfach einzusetzenden Konvertierungswerkzeugs ist streng modular. Das interne Dokumentmodell wird auf interne Adapter-Schnittstellen abgebildet, die bei Bedarf um weitere neue Ein- und Ausgabeformate erweitert werden können.

Eine weitere Stärke ist die AFP2web Kommunikationsschicht und deren Schnittstellenadapter. Viele Kunden nutzen täglich AFP2web mit der Auswahl eines beliebigen Kanals für Dokumenten-Eingabe oder -Ausgabe: Daten-EA, speicherresidente Datenströme, entfernte Dateien über FTP, proprietäre Datenbanken und bei Bedarf auch weitere Optionen.

Die folgende Abbildung zeigt die interne Modulstruktur von AFP2web:

AFP2web Interne Modulstruktur

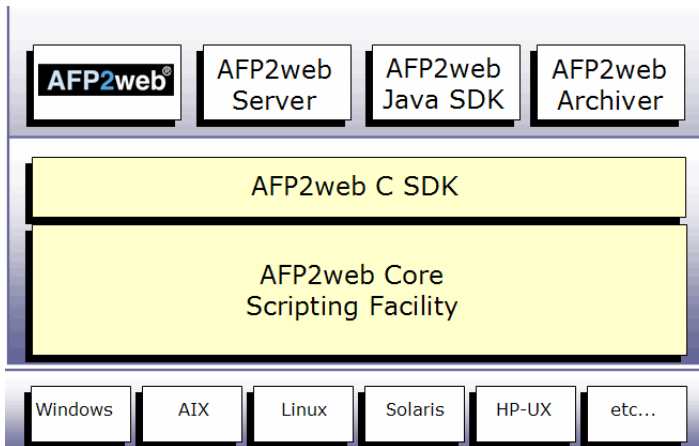


Methoden zur Integration von AFP2web

AFP2web bietet eine mächtige Basisfunktionalität, die für die jeweilige Anwendung zu konfigurieren ist. Die Basisversion von AFP2web starten Sie durch Aufruf über die Kommandozeile als Konsolenanwendung zur Verarbeitung von AFP Spooldateien. Die AFP2web Scripting Facility ist die Erweiterungsoption, mit der Sie Verarbeitungsregeln formulieren, und aus der heraus Sie weitere externe Programme starten. AFP2web und die Scripting Facility bieten eine intelligente und flexible Kontrolle der Dokumentkonvertierung.

Die folgende Abbildung zeigt wie man AFP2web Komponenten zu AFP2web Produkten kombinieren kann:

Komponenten zur Erstellung von AFP2web Produkte



Optional erhältlich ist ein Software Development Kit (SDK) zur Integration von AFP2web in bestehende Dokument oder Content Management Lösungen zur On-Demand Dokumentkonvertierung. Das AFP2web SDK beinhaltet eine AFP2web Dokument Management System (DMS) Application Programming Interface (API). Die Funktionen der AFP2web DMS API ermöglichen die Ein- und Ausgabe in Form von Datenströmen in speicherresidenten Pufferbereichen (afp2web) oder als Dateien (afp2web_f). Die Ein- und Ausgabe über Datenströme wird typischerweise für die Echtzeitkonvertierung einzelner Dokumente verwendet. Die Ein- und Ausgabe über Dateien ist der Normalfall für die Konvertierung von Massendruckdaten in Spooldateien.

Eine JNI Schnittstelle ermöglicht die einfache Integration in eine J2EE-Server-Umgebung wie IBM WebSphere.

1.2 AFP2web Anwendungsszenarios

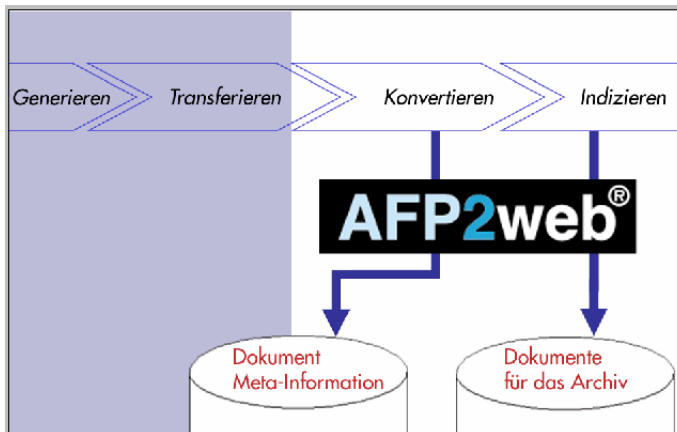
Typische Anwendungsszenarios mit AFP2web sind:

- Dokument-Archivierung und Index-Aktualisierung
- On-the-fly Konvertierung für PDF On-Demand
- AFP Viewer für Workstations
- Dokument-Versand per Post (Print), E-Mail oder Fax

Dokument-Archivierung und Index-Aktualisierung

Dieses Szenario beschreibt die automatisierte Konvertierung, Indizierung und Archivierung von Dokumenten. Der Prozess läuft im Stapelbetrieb und verarbeitet Massendruckdaten. Die Konvertierung muss problemlos und effizient ablaufen und ein Ergebnis mit hoher Qualität liefern.

Szenario: Dokument-Archivierung und Index-Aktualisierung



Der Prozess für die Dokumentarchivierung ist ein Workflow, der die Schnittstelle AFP2web für die Konvertierung und Archivierung von Dokumenten nutzt. Die Schritte, die der Prozess beinhaltet, sind:

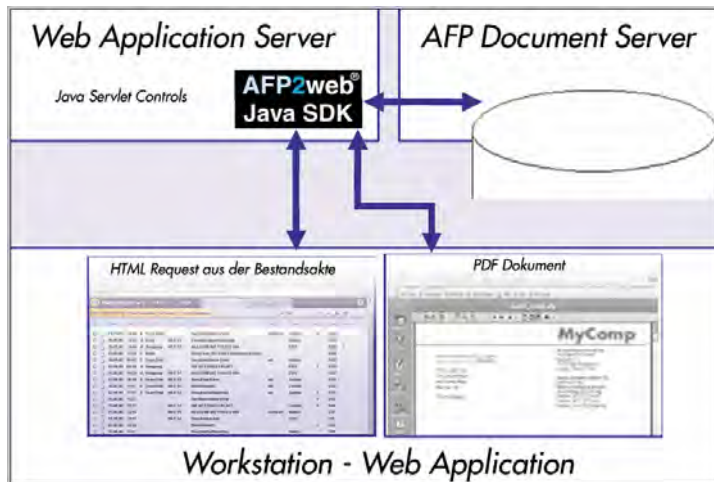
1. Generierung von AFP Daten
2. Transfer von AFP Daten (via FTP)
3. AFP2web Dokument-Konvertierung (zu PDF, zu einem der Rasterformate wie TIFF oder zu XML)
4. Dokument-Archivierung auf dem Dokument-Server und Ermittlung einer technischen Dokument ID
5. Aktualisierung der Index-Datenbank mit Dokument-Metainformation (von AFP2web zur Verfügung gestellt) und mit der Zuordnung zu der technischen Dokument ID

AFP2web liefert nicht nur ein exzellentes Konvertierungsergebnis. AFP2web kann zugleich als Extrahierungswerkzeug für Standard AFP Indexdaten arbeiten. Darüber hinaus kann die AFP2web Scripting Facility zur Verfeinerung der extrahierten Daten für jeden Typ von Information im Dokument benützt werden. Mit der Scripting Facility ist es auch möglich, Prozesse für die Folgeverarbeitung anzustoßen, z. B. in dokumentorientierten Workflows.

On-the-fly Konvertierung für PDF On-Demand

Ziel ist die Anzeige einzelner AFP Dokumente auf Anforderung in einem Web-Browser. Das erwünschte Format ist PDF mit einer originalgetreuen Wiedergabe des ursprünglichen AFP Dokuments.

Szenario: On-the-Fly Konvertierung für PDF On-Demand

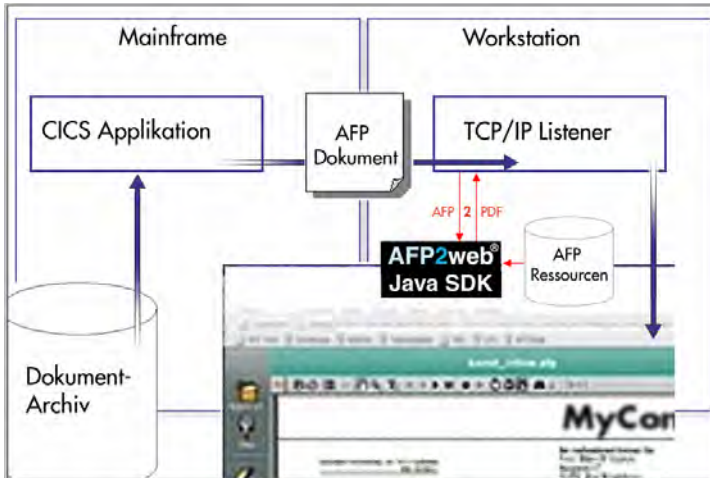


AFP2web wird als Komponente beispielsweise in einer dreistufigen Anwendungsarchitektur eingesetzt, typischerweise in einer Internet-Anwendung. Unter der Kontrolle eines Java Servlets liest AFP2web ein AFP Dokument aus dem Eingabepuffer, konvertiert es und schreibt es als PDF Dokument in den Ausgabepuffer. Durch die Vermeidung einer Datei Ein- und Ausgabe und die Verwendung von speicherresidenten Datenströmen ist AFP2web auf Schnelligkeit hin optimiert. Die Effizienz und hohe Qualität der Konvertierung ermöglicht die Online Betrachtung und schnelle Bearbeitung eines Dokumentes, auch in entfernten Büros.

AFP Viewer für Workstations

Ein Dokument im proprietären AFP Format, welches auf Anforderung aus dem Archiv kommt, wird für die sofortige PDF-Anzeige in einem Web Browser benötigt.

Szenario: AFP Viewer für Workstations



Mit der AFP2web Java Source Development Kit implementieren Sie AFP2web als unabhängigen Server für eingehende Konvertierungsanforderungen. Durch die Kombination mit einem beliebigen PDF oder TIFF Viewer wird AFP2web zu einem idealen AFP Viewer für die Darstellung von AFP Dokumenten. AFP2web liefert effiziente, qualitativ hochwertige Konvertierungsergebnisse.

Dokument-Versand per Post (Print), E-Mail, oder Fax

Die Konvertierung von AFP Dokumenten zu PDF oder TIFF und der Versand der Dokumente per Post, E-Mail oder Fax.

Wählen Sie das Ausgabeformat (z.B. PDF oder TIFF) das Sie benötigen und:

- Verwenden Sie einen E-Mail oder Fax-Server für den schnellen Versand von AFP Daten.
- Kombinieren Sie AFP2web mit einem Druckertreiber und drucken Sie AFP Dokumente farbig. Und das gelingt selbst auf einem Drucker ohne AFP-Unterstützung.

Kapitel 2: AFP2web Benutzerhandbuch

Dieses Kapitel enthält eine Anleitung zur

- Installation
- Systemanpassung für AFP2web

Die Anleitung zur Verwendung von AFP2web beschreibt generell

- den Schnelleinstieg mit Hilfe der Demo Dateien
- das Starten und Beenden von AFP2web
- die Nutzung bestimmter Merkmale von AFP2web
- AFP2web Fehlermeldungen und Log-Dateien für die Problembhebung

2.1 AFP2web installieren

Systemvoraussetzungen

Hardware und Betriebssystem Voraussetzungen

AFP2web kann auf einem dieser Betriebssysteme installiert werden:

- Windows NT/2000/2003 mit einem x86 Prozessor
- Linux mit einem x86 Prozessor
- AIX 5.1 32-bit RS/6000 PowerPC
- Sun Solaris 7 (SunOS 5.7) 32-bit auf einem Ultra-250 Sparc
- HP-UX 11.i 64-bit C3600a PA-RISC
- zLinux SuSE SLES7

Als Hauptspeicher erforderlich ist:

- wenigstens 512 MB RAM

Weitere Systemvoraussetzungen

Für die Konvertierung zu einem der Rasterformate (TIFF, JPEG) auf UNIX ist X-Windows erforderlich. Wenn kein XServer verfügbar ist, so kann alternativ Xvfb oder VNC verwendet werden.

AFP2web liefert für bestimmte Betriebssysteme (Prozessoren) eine Perl Engine mit. Wenn Sie Ihre eigene Perl Umgebung verwenden müssen, bitte installieren Sie Perl 5.8.7.

Die mit AFP2web mitgelieferte Perl Engines sind:

- Windows (Intel): Perl 5.8.7
- Aix 5.1 (RISC): Perl 5.8.7
- Solaris 2.7 (SPARC): Perl 5.8.7
- Suse Linux 9.1 (Intel): Perl 5.8.7
- HP UX 64: Perl 5.8.7

AFP2web wurde kompiliert auf HP-UX mit dem GNU-Compiler (V3.4.2) und nicht mit dem HP-Compiler (aCC).

Festplattenspeicher

Sorgen Sie dafür, dass Sie genügend Festplattenspeicher für Ihre Ein- und Ausgabedateien bereitstellen. Für die Schätzung des Speicherbedarfs mögen die folgenden Zahlen helfen:

- Eine resultierende TIFF-Datei ist um den Faktor 10 größer als das entsprechende Dokument im AFP-Datenstrom. Dies gilt für die G4-Komprimierung. Rechnen Sie mit dem Faktor 40 für eine gepackte TIFF-Datei oder mit bis zu Faktor 100 für die Option „Nicht-komprimiert“.
- Eine resultierende PDF-Datei ist etwa 1 bis 2 mal größer als das entsprechende Dokument im AFP Datenstrom. Dieser Wert ist abhängig von der Struktur eines AFP Dokuments.
- Verwenden Sie den Parameter **PDFDocLimits** der INI Datei, um die Speichergrenze zu erhöhen. PDFDocLimits kann die Performanz beeinflussen. Mit der Kommandozeilenoption **DOC_COLD** (Scripting Facility) wird Speicher für jedes resultierende PDF Dokument belegt. PDFDocLimits sollte daher auf den möglichst kleinsten Wert gesetzt sein, um die Performanz nicht zu beeinträchtigen. Mit der Kommandozeilenoption **-DOC** erfolgt die Reservierung von Speicher nur einmal, eine Änderung von PDFDocLimits hat daher keine Auswirkung auf die Performanz.
- Für jede in PDF eingebettete Schrift vergrößert sich diese PDF Datei im Durchschnitt um 30 KB bis 50 KB.
- Wenn Sie die Kommandozeilenoption **-sod** für Included Objects verwenden, so schreibt AFP2web diese Objekte auf Festplatte, um die Verarbeitung und die Objektdaten zu optimieren. In diesem Fall sorgen Sie bitte für weitere Festplattenkapazität.
- Log-Dateien von AFP2web können sehr groß werden. Die Performanz von AFP2web wird beeinträchtigt, wenn während der Konvertierung Log-Dateien erstellt werden. Log-Dateien werden normalerweise nur während der Entwicklung und nicht in einer Produktionsumgebung genutzt.

Schriften für die Konvertierung

Wenn Sie dies in der mapping.def als einen Schriftverarbeitungsmodus angeben oder wenn AFP2web die AFP Schrift nicht finden kann, dann ersetzt AFP2web diejenigen AFP-spezifischen Schriften mit Schriften, die Sie in den Konfigurationsdateien angeben haben. Die Anforderungen sind:

Für ein Rasterformat (TIFF, JPEG, PNG, AFP): Die Schrift sollte im Windows bzw. Unix Betriebssystem installiert sein.

Hinweis: In der früheren Version von AFP2web, musste eine Schrift für die Konvertierung nach TIFF im System installiert sein. AFP2web Version 3.x wird stattdessen eine erforderliche Schrift zur Laufzeit installieren und am Prozessende wieder deinstallieren.

Für PDF: AFP2web referenziert auf oder bettet Schriften ein.

Installation unter Windows

Vorbereitung: Es wird empfohlen, dass Sie die Konfigurationsdateien Ihrer letzten AFP2web Installation sichern. Somit können Sie frühere Anpassungen in den neuen Konfigurationsdateien neu definieren. Wenn Sie unseren Installationsempfehlungen gefolgt sind, so sind folgende Dateien bei der Datensicherung berücksichtigt:

- INI-Datei
- afp2web.pm (bzw. Ihre Skript-Module für die Scripting Facility)
- mapping.def
- *cp-Dateien

Sie installieren AFP2web unter Windows NT/2000/2003 wie folgt:

1. Starten Sie die Setup-Routine **setup.exe**

Die Setup-Routine ist eine selbstentpackende Datei, die alle AFP2web Dateien in ein gewähltes Zielverzeichnis abspeichert.

2. Wählen Sie das Zielverzeichnis, in das alle Dateien zu entpacken sind.

Klicken Sie hierzu auf die Schaltfläche [...].

- Markieren Sie die Option **Confirm overwrites**, wenn Sie von Fall zu Fall entscheiden möchten, ob eine vorhandene Datei überschrieben werden soll.
- Klicken Sie auf **OK**, um die Installation zu starten.

Der Dialog **Unpacking files** zeigt den Verlauf der Installation an.

Die Setup-Routine entpackt in wenigen Sekunden alle Dateien in den gewählten Zielordner und erstellt auch Unterverzeichnisse. Sie können mit **Cancel** den Vorgang jederzeit abbrechen.

3. Sollte der Dialog **File exists, overwrite?** erscheinen, dann wählen Sie eine der Optionen:

Ja Vorhandene Datei überschreiben

Nein Vorhandene Datei nicht überschreiben

Abbrechen Installation an dieser Stelle abbrechen.

Installation unter Unix

Unter UNIX entpacken Sie die Datei afp2web.tgz.

1. Geben Sie unter Solaris zwei Kommandos ein:

- `gzip -d afp2web.tgz`

- tar xvf afp2web.tar

2. Ansonsten geben Sie das Kommando ein: tar xvzf afp2web.tgz

Das Installationsergebnis

Die folgenden Dateien werden für AFP2web installiert

<i>Verzeichnis</i>	<i>Dateien</i>	<i>Beschreibung</i>
(Zielverzeichnis)	afp2web.exe (Windows) afp2web (Unix)	Das AFP2web Programm
	afp2web.ini	Konfigurationsdatei mit allgemeinen Einstellungen für AFP2web
	demo.bat (Windows) demo (Unix)	Batch-Datei mit Beispielen für den Programmaufruf
	history.txt	Änderungs-Historie
	license.txt	Lizenzinformationen
	quickstart.txt	Kurzanleitung für die Verwendung von demo.bat (Windows) oder demo (Unix)
	readme.txt	Letzte Hinweise
	afp2web.pm	Muster Skript-Modul für die AFP2web Scripting Facility
	*.pm files	Scripting Facility Beispiele
	Perl*.dll	Perl DLL für Ihr Betriebssystem (wenn vorhanden)
a2w	Perl module (*.pm)	PERL Packages mit der AFP2web Scripting Facility
afpcp	Code Pages (*.cp)	ASCII Code Page Dateien
	mapping.def	Konfigurationsdatei. Für die Zuordnung von AFP-spezifischen Schriften zu Ersatzschriften.
	gcgid2unicode.def	Default Unicode Zuordnungen.
afpcp\mfm		(obsolet)
doc	afp2web_de.pdf afp2web_en.pdf	AFP2web Benutzerhandbuch (en=Englisch, de=Deutsch)

<i>Verzeichnis</i>	<i>Dateien</i>	<i>Beschreibung</i>
log		Zielverzeichnis für die Protokolldateien (Log-Dateien)
pdf		Zielverzeichnis für konvertierte Dateien (Hinweis: für das afp2xml.pm Beispiel sind DTDs vorhanden.)
extfont	*.pfb, *.pfm, *.ttf files	Verzeichnis für weitere Adobe Type 1 und TrueType Schriften für die Konvertierung nach PDF und TIFF.
samples		Beispiele
samples\resource		Ressourcen für die Beispiele
XML	Writer.pm	Ein Perl Modul für das Skriptbeispiel afp2xml.pm

AFP2web deinstallieren

Um AFP2web zu deinstallieren, löschen Sie alle Dateien und Verzeichnisse, die von der AFP2web Installation erstellt wurden.

2.2 AFP2web konfigurieren

Überblick

Zur Konfiguration von AFP2web gehören das Setzen von Parametern und die Bereitstellung von Ressourcen:

- *"Parameter in der INI-Datei festlegen" auf Seite 44*
- *"Musterdokumente konvertieren und die Konfiguration anpassen" auf Seite 44*
- *"AFP Ressourcen festlegen" auf Seite 45*

Parameter in der INI-Datei festlegen

Die Konfigurationsdatei **afp2web.ini (INI-Datei)** legt global die Parameter von AFP2web fest. Hierzu gehören Konvertierungsoptionen, Ausgabepfade, Pfade zu AFP Ressourcen und weitere Verarbeitungsoptionen. Eine detaillierte Beschreibung der Parameter finden Sie unter *"Parameter der afp2web.ini" auf Seite 119*.

Standardmäßig sucht AFP2web im aktuellen Verzeichnis nach einer INI-Datei mit dem Dateinamen afp2web.ini. Sie können eine andere INI-Datei verwenden, wenn Sie eine der folgenden Kommandozeilenoptionen verwenden:

-if	Pfad und Dateiname der INI-Datei
-ip	Pfad zur afp2web.ini

Die meisten Parameter der INI-Datei lassen sich überschreiben, wenn Sie die entsprechenden Optionen in der Kommandozeile mitgeben. Diese Optionen sind unter *"AFP2web Kommandozeilenoptionen" auf Seite 145* beschrieben.

Musterdokumente konvertieren und die Konfiguration anpassen

Sie prüfen die Installation und die Einstellungen der Konfigurationsdateien, indem Sie zur Probe einige Musterdokumente konvertieren. Überprüfen Sie das Konvertierungsergebnis, die Meldungen der LOG-Datei und korrigieren, falls erforderlich, die Konfigurationseinstellungen.

In der nachfolgenden Beschreibung geben wir Ihnen einige Hinweise worauf Sie achten sollten.

Die AFP Spooldatei (Host Dokumente) binär herunterladen

Wenn Sie Ihre AFP-Dokumente und die dazugehörigen Ressourcen vom Host herunterladen, achten Sie bitte darauf, dass Sie diese binär herunterladen. (Bitte keine ASCII Umsetzung und keine Zeilenvorschübe hinzufügen!)

Konvertieren Sie die AFP Spooldatei und überprüfen Sie die LOG-Datei

Starten Sie das Programm afp2web.exe und konvertieren Sie Ihre Musterdokumente.

Eine Kurzanleitung wie AFP2web gestartet wird finden Sie unter ["Der Schnelleinstieg" auf Seite 49](#). Benützen Sie die beschriebene Demo Stapeldatei.

AFP2web erzeugt die folgenden Dateien:

- Das konvertierte Dokument, bzw. Dokumente oder, wenn die Konvertierung nicht möglich ist, eine Textdatei mit Fehlermeldungen.
- eine LOG-Datei, wenn dies aktiviert ist.

Wir empfehlen die Verwendung der LOG-Datei wenn Sie AFP2web ausführen. Die Ausgabe von LOG-Meldungen wird AFP2web wesentlich verlangsamen, ist aber in diesem Stadium notwendig. Um beispielsweise Informationen über die Verwendung von Schriften bei der Konvertierung zu erhalten, verwenden Sie den INI-Parameter **LoggingLevel=FONT** (oder die Kommandozeilenoption **-lf**).

Überprüfen Sie das konvertierte Dokument und gegebenenfalls die Systemmeldungen, um Probleme bei der Konvertierung zu erkennen. Eine fehlende Ressource oder Schrift ist möglicherweise die Ursache für ein unbefriedigendes Konvertierungsergebnis.

Beispiele für mögliche Probleme:

- Eine Ressource fehlt. In diesem Falle stellen Sie sicher, dass AFP2web die Ressource an der Stelle findet, die Sie der INI-Datei (oder als Kommandozeilenoption beim Aufruf von AFP2web) vorgeben.
- Verwenden Sie eine Ersatzschrift (während der Schriftverarbeitungsmodi font substitution/referencing), so kann die LOG-Datei auf das Fehlen einer Ersatzschrift hinweisen. In diesem Falle überprüfen Sie die mapping.def auf die korrekte Schriftzuordnung. Des Weiteren muss die Ersatzschrift im System verfügbar sein, d.h. die Schrift im System sollte installiert (für die Ausgabe in ein Rasterformat wie TIFF) bzw. im vorgegebenen Verzeichnis (für PDF) vorhanden sein.

Weitere Informationen hierzu finden Sie unter ["Die LOG-Datei nutzen" auf Seite 64](#) oder ["Die Ausgabe von Schriftinformationen in der LOG-Datei" auf Seite 77](#).

AFP Ressourcen festlegen

AFP Ressourcen (z.B. Fonts, Formdefs, Overlays, Page Segments) können inline im AFP Datenstrom mitgegeben werden. Wenn nicht, so sucht AFP2web nach externen Ressourcen in den Verzeichnissen, die Sie in der Konfiguration festlegen.

Hinweis: Der INI Parameter **CodePage** (bzw. die Kommandozeilenoption **-dcp**) definiert den Standard Code Page für die Umsetzung von Ressourcennamen, NOPs und Indizes im AFP Spool zu ASCII.

Normalerweise ist eine Konfiguration für die AFP2web Schriftverwaltung nicht erforderlich. Ist die AFP Schrift als Ressource inline oder extern verfügbar, so bettet AFP2web die Schrift entweder für die Ausgabe (bei PDF) ein oder schreibt die Schrift in das auszu-

gebende Rasterformat (z.B. für die Ausgabe nach TIFF). In beiden Fällen wird die AFP Schrift verwendet und die Konvertierung erzeugt Dokumente in originalgetreuer Darstellung.

Weitere Informationen über die Verwendung von Schriften, Code Pages, Konfigurationsdateien finden Sie unter *"Die Behandlung von Schrift bei der Konvertierung"* auf Seite 67.

Pfad zu externen AFP Ressourcen

Es gibt mehrere Möglichkeiten, externe AFP Ressourcen zur Verfügung zu stellen. Zum Beispiel kann durch die Eingabe der Kommandozeilenoption **-rf** bestimmt werden, eine spezielle Ressource zu benutzen, oder Sie geben in der INI-Datei die Pfade zu den speziellen Ressourcentypen an.

Mit allen Parametern außer **ResPath (-rp)** und der Kommandozeilenoption **-rf** können Sie eine mit kommagetrennte Auflistung von Pfaden angeben. Die Parameter für AFP Ressourcen sind:

<i>INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Art der AFP Ressource</i>
ResPath	-rp	Alle Arten von Ressourcen
	-rf	Alle Arten von Ressourcen in einer Datei
AFPFontPath	-ap	AFP Fonts
OverlayPath	-ovp	AFP Overlays
PageSegmentPath	-psp	Page Segments
FormDefPath	-fdp	Formdefs

Die Suche nach Ressourcen

AFP2web sucht nach externen Ressourcen in der folgenden Reihenfolge:

1. Im Verzeichnis für die Art der Ressource (standardmäßig in samples/resource)
2. Im Verzeichnis von **ResPath** (standardmäßig in samples/resource)
3. Im Verzeichnis der Eingabedateien
4. Im Standard Verzeichnis "samples/resource"

Die Suche nach Nicht-AFP Ressourcen

AFP2web sucht auch nach Ressourcen, die nicht AFP spezifisch sind, beispielsweise nach TIFF/JPEG Dateien, die in IOBs referenziert werden. Der Suchweg für solche Ressourcen ist:

1. Ressourcenpfad (definiert im **ResPath** INI-Parameter)
2. Pfad der Eingabedatei
3. Im aktuellen Verzeichnis

Optional: Die Definition von Dateierweiterungen für AFP Ressourcen

Die INI-Datei bietet die Möglichkeit, Dateinamenserweiterungen für die folgenden Arten von Ressourcen zu definieren:

FormdefExt	Dateinamenserweiterung für Dateien mit Formdefs
OverlayExt	Dateinamenserweiterung für Overlays
PageSegExt	Dateinamenserweiterung für Page Segments
CharSetExt	Dateinamenserweiterung für Character Sets
CodePageExt	Dateinamenserweiterung für Code Pages
CodedFontExt	Dateinamenserweiterung für Coded Fonts

Der AFP2web Prozess während der Suche nach einer Ressource

AFP2web benötigt keine Dateinamenserweiterungen für eine Ressource. Der Ablauf ist:

1. AFP2web richtet sich nach den Einstellungen (den vorgegebenen oder Benutzer-spezifischen Erweiterungen) in der INI-Datei. Die Suchweg ist wie zuvor beschrieben.
2. Kann AFP2web die Ressource in Schritt 1 nicht finden und ist die Option **Strict** aktiviert, so endet AFP2web.
3. Kann AFP2web die Ressource in Schritt 1 nicht finden und ist die Option **Strict** deaktiviert, so sucht AFP2web wiederum auf dem vorgegebenen Suchweg nach der gleichnamigen Ressource ohne Rücksicht auf eine etwa vorhandene Dateinamenserweiterung. Die erste Ressource, die AFP2web findet, wird genommen.

Wichtig: Da AFP2web Dateinamen, die mit dem Ressourcennamen beginnen, auswählt, überprüfen Sie sicherheitshalber, ob die korrekte Ressource gefunden wird.

Beispiel: Gesucht wird nach der Ressource "C1H20000". AFP2web sucht nach "C1H20000*". Die folgenden Ressourcen sind gültig: C1H20000, C1H20000.chs, C1H20000.240, C1H20000_ABC.def. Wenn mehr als eine Ressource dem Muster entspricht, so wird der erste Treffer genommen.

Ist in der INI-Datei die Dateinamenserweiterung "chs" angegeben, so findet AFP2web die Ressource in Schritt 1: "C1H20000.chs".

Ist in der INI-Datei als Dateinamenserweiterung "*" oder nichts angegeben, so findet AFP2web die Ressource erst in Schritt 3: "C1H20000".

Formdefs angeben

Wie zuvor in diesem Abschnitt beschrieben, können AFP Ressourcen (wie Formdefs, Overlays, Page Segments) inline im AFP Datenstrom mitgegeben werden. Wenn nicht, dann sucht AFP2web in den angegebenen Verzeichnissen nach einer externen Ressource. Beispiel: Sie verwenden in der Kommandozeile eine der Kommandozeilenoptionen **-fd** oder **-rf**, um die Ressourcendatei mit der Formdef anzugeben.

Ein Formdef ist eine AFP Ressource, die Medium Maps und/oder Copy Groups enthält. Die Formdef selber wird aber nicht im AFP Datenstrom referenziert. In der INI-Datei definieren Sie im **Abschnitt [FORMDEF]** die Zuordnung von Medium Maps zu Formdefs. Wenn Sie die Formdef nicht explizit mit der Kommandozeilenoption **-fd** angeben, so kann AFP2web die Angaben in der INI-Datei nutzen. Die Formdef selber muss als externe Ressource verfügbar sein.

2.3 AFP2web anwenden

Der Schnelleinstieg

So einfach können Sie Ihre AFP-Dokumente mit AFP2web konvertieren:

1. Achten Sie darauf, dass Sie Ihre AFP-Dokumente und die dazugehörigen Ressourcen vom Host binär herunterladen.
2. Kopieren Sie Ihre AFP-Dokumente in das Verzeichnis **samples/**.
3. Kopieren Sie die Ressourcen in das Unterverzeichnis **samples/resource/**.
4. Bearbeiten Sie **demo.bat** (Windows), bzw. **./demo** (Unix) und geben Sie den Namen der zu konvertierenden AFP Spooldatei an.
5. Starten Sie **demo.bat** (Windows), bzw. **./demo** (Unix)
6. Überprüfen Sie Ihre Resultate im Verzeichnis **pdf/**.
7. Wiederholen Sie Schritt 1 bis 6 für jede AFP Spooldatei.

Hinweis: *Treten Probleme auf, kontaktieren Sie bitte afp2web@maas.de.*

AFP2web in der Kommandozeile starten, stoppen

AFP2web starten

Sie starten das Programm `afp2web` über die Kommandozeile als Anwendung in der Konsole. Hierzu geben Sie als Kommandozeilenoption den Namen der zu konvertierenden Datei ein:

In Windows: `afp2web.exe [-options ...] inputfile`

In UNIX: `./afp2web [-options ...] inputfile`

Für Inputfile sind Platzhalter (sog. „wildcards“) erlaubt, z.B. `"*.afp"` für alle Dateien, die mit `".afp"` enden.

Eine detaillierte Beschreibung weiterer Optionen finden Sie unter *"AFP2web Kommandozeilenoptionen"* auf Seite 145.

AFP2web stoppen

In Windows drücken Sie die Tastenkombination **CTRL+C** in der Eingabeaufforderung oder stoppen Sie den Prozess mit dem Task Manager.

In Unix beenden Sie den AFP2web Prozess mit dem Befehl **kill**.

Die Konvertierung steuern

Sie definieren die Parameter der INI-Datei, um die Formate und die Art der Ausgabedokumente zu bestimmen.

Der Parameter **InputFormat** definiert das zu konvertierende Eingabeformat, wie z.B.

AFP	AFP Dokument
TIFFIN	TIFF Dokument

Der Parameter **OutputFormat** legt das Zielformat für die Konvertierung fest, wie z. B.

ASCII	ASCII Format mit Anpassungen in den Zeilen/Spalten-Positionierungen
JPEG	JPEG Format
PDF	PDF Format
PNG	Portable Network Graphic
TIFF	TIFF Format

Der Parameter **FileCreationMode** steuert die Trennung oder die Zusammenfügung von Dokumenten aus der Spooldatei, die Indexerstellung und die Aktivierung der Scripting Facility:

ALL	Eine Ausgabedatei für jede Eingabedatei.
DOC	Mehrere Ausgabedateien: Jeweils eine Ausgabedatei für jedes AFP-Dokument.
DOC_MERGE	Mehrere Dateien zu einer Ausgabedatei zusammenführen. Gilt für Rasterformat (TIFF) nach PDF und für AFP als Eingabeformat.
DOC_INDEX	Jeweils eine Indexdatei und eine Ausgabedatei pro Einheit laut Index Detailinformationen.
DOC_COLD	Dokumente trennen und Dokumentelemente verarbeiten — mit der AFP2web Scripting Facility.
PAGE	Eine Datei pro Seite in einem AFP-Dokument.

Dokumente für die Konvertierung auswählen

Zur Verarbeitung einer AFP Spooldatei gehört die Konvertierung einzelner AFP Dokumente und die Verarbeitung der Indexdaten. Die AFP Indexdaten dienen dazu, Beginn und Ende der einzelnen AFP Dokumente zu erkennen. Eine Alternative ist es, die Dokumenterkennung mit Hilfe der Scripting Facility zu formulieren. Wenn Sie die AFP2web Standardverarbeitung für Indexdaten wünschen, wählen Sie **FileCreationMode=DOC_INDEX** (Siehe auch ["AFP Indexelemente verarbeiten" auf Seite 54](#)). Wenn Sie die Dokumenttrennung und die Verarbeitung der Indexdaten mit Hilfe der Scripting Facility ausführen wollen, wählen Sie **FileCreationMode=DOC_COLD** (Siehe auch ["Grundlagen der AFP2web Scripting Facility" auf Seite 91](#)).

Sie können AFP2web anweisen, dass die Verarbeitung in der Spooldatei an einem bestimmten Dokument gestartet oder beendet werden soll. AFP2web parst die komplette Spooldatei bis zum angegebenen Dokument. Diese Option setzen Sie für besondere Fälle wie zum Beispiel zur Fortsetzung der Konvertierung nach einem Abbruch oder zur Fehlerbehebung ein.

Tipp: Mit der Kommandozeilenoption **-p** erzeugen Sie die Ausgabe eines Protokolls. Das Protokoll ist eine hilfreiche Referenz für die Dokumentenauswahl mit Informationen über die Konvertierung. In diesem Protokoll ist eine Liste der verarbeiteten Dokumente und deren Dokumenten-IDs. Das Protokoll finden Sie im Ordner für Ausgabedokumente (festgelegt durch den INI Parameter **OutputFilePath**, bzw. die Kommandozeilenoption **-op**). Der Dateiname des Protokolls setzt sich zusammen aus: „protocol“ + Zeitstempel + „.txt“.

Beispiel: Ausschnitt aus einem Protokoll mit Dokument IDs

```
Running AFP2Web Version 3.0rc1 [Built for Windows NT/2000/2003 on
Dec 18 2005 at 13:48:50] at 21:34:06 on 21 December 2005 using :
-q -PDF -c -p -DOC_INDEX:XML -op:pdf\xml samples\insure.afp
```

```
Processing Spool File: samples\insure.afp
[21:34:09] Processing Document Id : 1, pdf/xml/insure_s1mc.1.pdf
: Ok
[21:34:11] Processing Document Id : 2, pdf/xml/insure_s1mc.2.pdf
: Ok
[21:34:11] Processing Document Id : 3, pdf/xml/insure_s1mc.3.pdf
: Ok
[21:34:12] Processing Document Id : 4, pdf/xml/insure_s1mc.4.pdf
: Ok
[21:34:12] Processing Document Id : 5, pdf/xml/insure_s1mc.5.pdf
: Ok
```

```
AFP2Web Version 3.0rc1 [Built for Windows NT/2000/2003 on Dec 18
2005 at 13:48:50] ended at 21:34:13 on 21 December 2005, Elapsed
Time 00:00:07
```

Ein Fallbeispiel für die Verwendung des Parameters StartingDocument: Ein Fehler (wie Speicherplatzmangel oder eine fehlende Ressource) kann der Grund dafür sein, dass AFP2web die Konvertierung abbricht. In diesem Falle schauen Sie im Protokoll nach, welche Dokumente erfolgreich konvertiert wurden.

Nach Behebung eines Problems können Sie die Konvertierung ab einer bestimmten „Dokument Id“ fortsetzen, ohne die ganze Spooldatei erneut konvertieren zu müssen. Sie verwenden hierfür den INI-Parameter **StartingDocument**, bzw. die Kommandozeilenoption **-sd**. Die „Document Id“ entnehmen Sie bitte dem Protokoll. Mit **EndingDocument** (oder mit der Kommandozeilenoption legen Sie das letzte Dokument fest. So können Sie die Konvertierung auf eine begrenzte Auswahl von Dokumenten einschränken.

Die Parameter für die Verarbeitung von Spooldateien im Überblick:

<i>INI-Datei</i>	<i>Kommandozeilenoption</i>
FileCreationMode=DOC_INDEX oder FileCreationMode=DOC_COLD	-DOC_INDEX oder -DOC_COLD
Protocol=on	-p
StartingDocument	-sd
EndingDocument	-ed

Dokumentseiten für die Konvertierung auswählen

Sie können einen Bereich von Seiten innerhalb eines AFP Dokuments auswählen. Die Parameter hierfür sind:

<i>INI-Datei</i>	<i>Kommandozeilenoption</i>
StartingPage	-pp:[fp][-tp] (Verarbeite Seite von/bis)
EndingPage	

AFP Indexelemente verarbeiten

Im Folgenden beschreiben wir die Standardverarbeitung von AFP Indexdaten mit AFP2web. Wenn Sie eine maßgeschneiderte Lösung realisieren wollen, lesen Sie bitte die Einführung in die Scripting Facility.

Ist AFP2web mit dem Parameter **FileCreationMode=DOC_INDEX** konfiguriert, liest AFP2web Standard AFP Indizes aus dem AFP Datenstrom und gibt diese in einem der gewählten Formate aus.

AFP2web schreibt die Indexdaten in das Verzeichnis, das Sie mit dem Parameter **Output-FilePath**, bzw. mit dem Parameter **IndexPath** festlegen.

Werden die Standard AFP Indizes nicht im AFP Datenstrom, sondern in einer externen Datei mitgegeben, so geben Sie die externe Datei als weiteres Argument an AFP2web, als Kommandozeilenoption **-xf** mit an.

Sie legen das Format der auszugebenden Indexdaten mit dem Parameter **IndexFormat** fest. Möglich ist eines der Standardformate XML oder CSV (kommagetrennte Werte).

Ein Beispiel für das XML Format:

Standard XML Format für Indexdaten

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE Index SYSTEM "Index.dtd" >
<Index Doctype="PDF" DocName="pdf/xml/INSURE_s3k.0.pdf"
PageCount="3" Size="16726">
  <PageGroupIndex PageGroup="00000001">
    <Data>
      <Name>Insured</Name>
      <Value>Geoffrey R Stephens</Value>
    </Data>
    <Data>
      <Name>Policy</Name>
      <Value>324-1443255-11</Value>
    </Data>
  </PageGroupIndex>
  <PageIndex Page="00000001">
    <Data>
      <Name>Contents</Name>
      <Value>Disability Income Policy</Value>
    </Data>
  </PageIndex>
```

(weitere Zeilen hier nicht gezeigt...)

```
<PageIndex Page="00000003">
  <Data>
    <Name>Contents</Name>
    <Value>Definitions</Value>
  </Data>
</PageIndex>
</Index>
```

Wenn Sie CSV als Ausgabeformat für den Index festlegen, haben Sie die Möglichkeit, mit dem Parameter **IndexRecord** weitere Metainformationen pro Indexelement mit auszugeben. Informationen werden in der Reihenfolge ausgegeben, in der Sie diese als Schlüsselwörter im Parameter IndexRecord angeben. Zum Beispiel:

IndexRecord=PAGE_GROUP_NAME,
PAGE_NAME,PAGE_COUNT,FILE_NAME,FILE_TYPE,INDEX,FILE_SIZE

Ein Beispiel für die Ausgabe im CSV-Format:

Standard CSV Format für Indexdaten

```
PAGE_GROUP_NAME, PAGE_NAME, PAGE_COUNT, FILE_NAME, FILE_TYPE, INDEX, FILE_SIZE
00000001, , 3, pdf/csv/INSURE_s73.0.pdf, PDF, Insured=Geoffrey R Stephens, 16726
00000001, , 3, pdf/csv/INSURE_s73.0.pdf, PDF, Policy=324-1443255-11, 16726
, 00000001, 3, pdf/csv/INSURE_s73.0.pdf, PDF, Contents=Disability Income Policy, 16726
, 00000002, 3, pdf/csv/INSURE_s73.0.pdf, PDF, Contents=Policy Schedule, 16726
, 00000002, 3, pdf/csv/INSURE_s73.0.pdf, PDF, Contents=Policy Schedule, 16726
, 00000002, 3, pdf/csv/INSURE_s73.0.pdf, PDF, Contents=Table of Benefits, 16726
, 00000002, 3, pdf/csv/INSURE_s73.0.pdf, PDF, Contents=Additional Benefits, 16726
```

Die Parameter für die Erzeugung von Indexdateien im Überblick:

<i>INI-Datei</i>	<i>Kommandozeilenoption</i>
InputFormat=AFP	
OutputFormat=PDF	-PDF
FileCreationMode=DOC_INDEX	-DOC_INDEX
IndexPath	-xp -xf
IndexFormat=CSV XML	
IndexRecord=fieldlist	

Ausgabedokumente in Unterverzeichnisse ausgeben

Wenn eine große Anzahl von Dokumenten in einem Verzeichnis gespeichert werden, ist unter Umständen ein effizienter Zugriff auf das Dateisystem nicht mehr möglich.

Für diesen Fall bietet AFP2web die Kommandozeilenoption **-dc** (bzw. den INI-Parameter **DocumentCount**) an, die Sie zusammen mit der Option **DOC_COLD** oder **-DOC_INDEX** verwenden können. Mit der Kommandozeilenoption **-dc** erzeugen Sie weitere Unterverzeichnisse und legen die maximale Anzahl von Dokumenten pro Unterverzeichnis fest.

Beispiel: Die Kommandozeilenoption **-dc:500** zusammen mit **-doc_cold:xml** schränkt die Anzahl der zu verarbeitenden Dokumente auf 500 pro Unterverzeichnis ein. Das bedeutet: Sie bekommen jeweils 1000 Dateien pro Unterverzeichnis, das sind 500 Ausgabedokumente und 500 Indexdateien im XML Format.

Wichtig: AFP2web verhindert weder das Überschreiben bereits vorhandener Dateien in einem Verzeichnis, noch prüft es zu Beginn die Anzahl bereits vorhandener Dateien.

Wir empfehlen daher:

- Verwenden Sie AFP2web mit der Kommandozeilenoption **-dc** nur wenn Sie eine einzelne große Spooldatei verarbeiten.
- Verwenden Sie **-dc** nur, um neue Unterverzeichnisse für Ihre Ausgabe anzulegen, nicht um etwa vorhandene Unterverzeichnisse zu erweitern.

*Beispiel für die Ausgabe mit der Kommandozeilenoption **-dc:500** und **-doc_index**:*

```
Folder ... \pdf
<DIR>          00
<DIR>          01

Subfolder ... \pdf\01
insure_s1eo. 1. pdf
insure_s1eo. 1. xml
insure_s1eo. 2. pdf
insure_s1eo. 2. xml
insure_s1eo. 3. pdf
insure_s1eo. 3. xml
...
insure_s1eo. 500. pdf
insure_s1eo. 500. xml

Subfolder ... \pdf\01
insure_s1eo. 501. pdf
insure_s1eo. 501. xml
...
```

Die Verarbeitung von eingebetteten Objekten (Included Objects)

Nicht-AFP-Daten (in der Regel Rastergrafiken wie Logos oder Farbgrafiken) werden in AFP Objektbehälter, sog. Included Objects, eingebettet oder referenziert.

AFP2web optimiert das Konvertierungsergebnis:

- Farbige Grafiken werden im JPEG Format (JPEG ist ein verlustreiches Format) ausgegeben. Sie verwenden den INI Parameter **JPEGQuality** (oder die Kommandozeilenoption **-jq**), um die Qualität der Komprimierung auszuwählen.
- Schwarz/Weiß Grafiken werden zu TIFF G4 Format konvertiert.

Unterstützte Formate und wie AFP2web Objektdaten findet

Die in Included Objects eingebetteten Objektdaten können inline oder in externen Dateien vorliegen.

Damit AFP2web ein externes Objekt laden kann, muss AFP2web den Dateinamen und Dateityp kennen. Diese Information liest AFP2web aus dem AFP-Datenstrom.

Die Objekttypen, die AFP2web erkennt, und deren Dateierweiterungen sind:

<i>Objekttyp (Unterstützte Formate)</i>	<i>Dateierweiterung</i>
TIFF (G4, G3, LZW, Packbits, Uncompressed, JPG, CCI-TTRLE, CCITTRELEW, IT8CTPAD, IT8LW, IT8MP, IT8BL, PIXAR-FILM, PIXARLOG, DEFLATE, ADOBE_DEFLATE, THUNDERSCAN, DCS, NEXT, SGILOG, SGILOG24)	tif
JPEG	jpg
Windows Bitmap	bmp
PCX	pcx

Referenziert ein Objektbehälter ein externes Objekt, so setzt sich der Dateiname wie folgt zusammen:

<Objektname in SFI>.<Dateierweiterung>

Beispiel: Wenn der AFP-Datenstrom das Objekt MOC1 des Typs TIFF enthält, sucht AFP2web nach einer Datei mit dem Namen: MOC1.tif.

Nach der Ermittlung des Dateinamens sucht AFP2web nach einer externen AFP Ressource wie folgt:

1. im Ordner laut **ResPath** (standardmäßig samples/resource)
2. im Ordner der Eingabedateien
3. im aktuellen Verzeichnis

Optionen für die Optimierung

Standardmäßig lädt AFP2web Objektdaten in den Speicher. Das erhöht den Speicherbedarf erheblich. Es ist von daher nicht notwendig, nur selten genutzte große Grafiken in einer AFP Spooldatei ständig im Speicher zu halten.

Um die Konvertierung an dieser Stelle zu optimieren, bietet AFP2web den Parameter **SaveOCDataOnDisk** (bzw. die Kommandozeilenoption **-sod**). Mit dieser Einstellung werden eingebettete Objekte (Grafikdateien) temporär auf Festplatte geschrieben und nicht im Arbeitsspeicher geladen und verarbeitet.

Die temporären Dateien für die Daten im Objektbehälter werden in dem von **TempPath** festgelegten Pfad abgelegt. Wird TempPath nicht angegeben, so gilt der vom System vorgegebene Pfad für temporäre Dateien. Die entsprechende Kommandozeilenoption ist **-tp**.

Nach der Verarbeitung entfernt AFP2web die temporären Dateien in der Aufräumphase.

Hinweise zur Barcode Unterstützung

AFP2web unterstützt den Code 128 Barcode (Character Set B des Code 128).

AFP2web unterstützt ebenfalls den zweidimensionalen DataMatrix 2D Barcode wie folgt:

- 1. Quadratische Formate mit einem Datenbereich werden von AFP2web unterstützt. Quadratische Formate mit mehr als einem Datenbereich werden nicht unterstützt.
- 2. Rechteckige Formate werden von AFP2web unabhängig von der Anzahl der Datenbereiche unterstützt.

Die folgende Tabelle gibt in Detail an, welche Symbolgrößen des DataMatrix 2D Barcodes von AFP2web unterstützt werden. Die Zahl in Klammern steht für die Anzahl der Datenbereiche der jeweiligen Größe.

1a. Unterstützte rechteckige Symbolgrößen.	8x18 (1)
	8x32 (2)
	12x26 (1)
	12x36 (2)
	16x36 (2)
	16x48 (2)
2a. Unterstützte quadratische Symbolgrößen (Diese Größen weisen nur einen Datenbereich auf.)	10x10
	12x12
	14x14
	16x16
	18x18
	20x20
	22x22
	24x24
	26x26

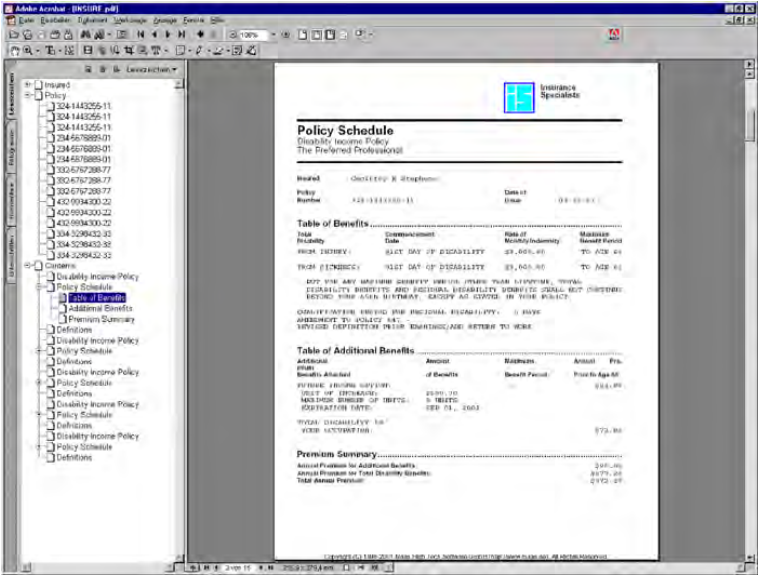
2b. Quadratische Symbolgrößen, die von AFP2web nicht unterstützt werden. (Diese Größen weisen mehr als einen Datenbereich auf.)	32x32 (4)
	36x36 (4)
	40x40 (4)
	44x44 (4)
	48x48 (4)
	52x52 (4)
	64x64 (16)
	72x72 (16)
	80x80 (16)
	88x88 (16)
	96x96 (16)
	104x104 (16)
	120x120 (32)
	132x132 (32)
	144x144 (32)

PDF Lesezeichen für Indexelemente anlegen

Mit **PDFBookmark** (bzw. mit der Kommandozeilenoption **-bm**) erzeugen Sie in den PDF Dateien Lesezeichen für Indexelemente.

Ein Beispiel für die erzeugten Lesezeichen in einer Darstellung durch den Acrobat Reader:

PDF Lesezeichen für Indexelemente



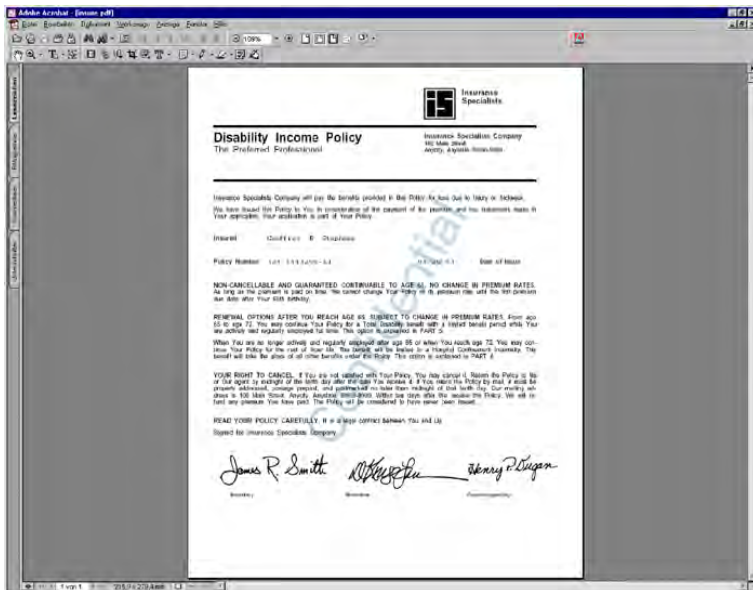
Die Parameter für die Erzeugung von PDF Lesezeichen:

<i>INI-Datei</i>	<i>Kommandozeilenoption</i>
InputFormat=AFP	
OutputFormat=PDF	-PDF
PDFBookmark	-bm

Einen Standardtext als Wasserzeichen unterlegen

Mit dem Parameter **Watermark** in der INI-Datei oder mit der Kommandozeilenoption – **wm** können Sie einen Text als Wasserzeichen dem Dokumententext hinterlegen. Ein Beispiel:

Text als Wasserzeichen im Hintergrund einer Dokumentenseite



Konvertierung abbrechen, wenn eine Ressource fehlt

Mit der Kommandozeilenoption **-strict** sorgen Sie dafür, dass der Ablauf gestoppt wird, wenn eine AFP-Ressource fehlt. Dies gilt für die Ressourcen vom Typ Page Segment, Overlay, Formdef, AFP Font und nicht-AFP-Ressourcen.

Diese Funktion ist für Sie nützlich, wenn Sie solche Fehler in der Konvertierung nicht zulassen dürfen.

Die Parameter im Überblick:

<i>INI-Datei</i>	<i>Kommandozeilenoption</i>
Strict=on	-Strict

Die LOG-Datei nutzen

Eine LOG-Datei erstellen

Name der LOG-Datei: Die LOG-Datei ist eine Textdatei mit Aufzeichnungen zur Konvertierung. Ist Logging aktiviert, so erstellt AFP2web für jede AFP Datei eine separate LOG-Datei. Die LOG-Datei hat als Dateiname den Namen und Erweiterung des Eingabe-Dokuments mit der zusätzlichen Dateierweiterung ".log". Beispiel: Die Eingabedatei INSURE.AFP wird konvertiert zu INSURE.PDF mit der dazugehörigen LOG-Datei: INSURE.AFP.log.

Den Pfad für LOG-Dateien festlegen: Sie legen den Ordner, in den LOG-Dateien generell geschrieben werden, mit dem INI-Parameter **LogPath** fest. Sie können den Parameter der INI-Datei übergehen, indem Sie mit der Kommandozeilenoption **-lp** einen anderen Ordner explizit angeben.

Die LOG-Datei für Schriftinformationen verwenden: Sie können AFP2web dazu veranlassen, eine LOG-Datei zu erstellen, in der informiert wird, wie AFP-spezifische Schriften durch Ersatzschriften ersetzt werden. Wir empfehlen Ihnen, die LOG-Datei ausschließlich für diesen Zweck zu verwenden. Sie erzeugen eine LOG-Datei mit Schriftinformationen entweder generell in der INI-Datei mit dem Parameter **LoggingFont=on** oder einmalig beim Programmaufruf mit der Kommandozeilenoption **-lf**.

Weitere Optionen: Sie können detaillierte Informationen in einer LOG-Datei protokollieren lassen. Diese Zusatzinformation ist für den AFP-Experten und für den AFP2web Support hilfreich. Die Möglichkeiten hierzu:

- Einen der Parameter der INI-Datei **Logging=on**, **LoggingLevel**, oder **ExceptionLoggingLevel**
- Die Kommandozeilenoption **-l**, oder **-ll**

Informationen über Ausnahmebedingungen

Standardmäßig gibt AFP2web Meldungen für Ausnahmebedingungen ohne weitere Informationen aus. Sie können weitere Informationen ausgeben mit Hilfe des Parameters **ExceptionLoggingLevel**.

Struktur der LOG-Datei

Die LOG-Datei besteht aus zwei Teilen:

Parsing-Teil	in dem AFP-Parameter aufgeführt werden. Hier finden Sie die AFP-Parameter für Schriften und auch Meldungen, ob Ressourcen gefunden wurden oder nicht. Der Parsing-Teil endet mit der Meldung „end-of-file“. Dieser Teil ist in erster Linie als Information für den AFP2web Support nützlich.
Builder-Teil	in dem Schriften für die Konvertierung aufgeführt werden. Wenn Sie die LOG-Datei nutzen wollen, dann empfehlen wir, diesen Teil zu nutzen. Hier können Sie am einfachsten feststellen, wie AFP-spezifische Schriften durch Ersatzschriften ersetzt wurden.

Das Format der LOG-Datei:

<i>Parsing-Teil Satzstruktur</i>	
Spalte 1 - 10:	Spalte 1 - 10: Datei-Offset des Structured Field Introducer (SFI) im Hex-Format. Der SFI kann als MO:DCA-Befehl verstanden werden.
Spalte 14 – 21:	Der SFI im Hex-Format.
nach Spalte 23:	Die Parameter, die im SFI enthalten sind.

<i>Builder-Teil Satzstruktur</i>	
PDF Seitennummer	Beispiel: Writing Page 1
Schrift (Character Set, Code Page)	Beispiel: ==>CS=C1H400B0, TF=HELVE-TICA LATIN1, W=B, S=12.00, RF, CP=T1GI0361(MD:2065.cp)
Verwendete Schrift und Text mit Angaben zu Textpositionen	Beispiel: -->UsedFont=C1H400B0, TF=F1, S=12.00, T3 @(8496,862)>Insurance< @(8496,1120)>Specialists<

Kapitel 3: Die Behandlung von Schrift bei der Konvertierung

In diesem Kapitel beschreiben wir die Verarbeitungsmodi für Schrift und zeigen einige Beispiele. Doch zuvor geben wir eine kurze Einführung in die Konfigurationsdateien, die sich auf die Verarbeitung von Schrift auswirken.

Hinweis: Wir bezeichnen Schriften in einem Eingabe-Dokument vereinfachend *“Eingabe-Schriften”* und Schriften in einem Ausgabe-Dokument als *“Ausgabe-Schriften”*.

Schriften sind entweder eingebettet, bzw. inline in den Dokumenten, oder sie werden als externe Schriften referenziert. Alle verwendeten Schriften müssen für AFP2web verfügbar sein.

3.1 Die Konfigurationsdateien und Voreinstellungen für Schrift

Die Konfigurationsdateien, die sich auf die Verarbeitung von Schrift auswirken:

- Die INI-Datei (afp2web.ini)
- Die Konfigurationsdatei für Schrift (mapping.def)
- Code Page Dateien (CP-Dateien)
- Standardannahmen bei der Zuordnung von Schrift

Kann AFP2web eine angegebene Schrift nicht finden, so greift AFP2web auf eine Ersatzschrift zurück, damit ein Konvertierungsergebnis gewährleistet bleibt.

Die INI-Datei (afp2web.ini)

Die INI-Datei steuert die Verarbeitung in AFP2web. Anstelle eines INI-Parameters kann auch die entsprechende Kommandozeilenoption verwendet werden.

Eine detaillierte Beschreibung der Parameter und Syntax finden Sie in der *"Gegenüberstellung: INI-Parameter, Kommandozeilenoptionen"* auf Seite 111 und *"Parameter der afp2web.ini"* auf Seite 119.

Die folgende Tabelle listet alle Parameter der AFP2web INI-Datei auf, die sich auf die Umsetzung von Schrift auswirken:

<i>INI Parameter</i>	<i>Kommandozeilenoption</i>
Parameter für die Ausgabe einer LOG-Datei:	
LoggingLevel=FONT	-ll:FONT oder -lf
LoggingLevel=ALL	-ll:ALL
LoggingLevel=FNTR	-ll:FNTR
Schalter für den Programmabbruch im Falle einer fehlenden Ressource:	
Strict= on	-Strict
Pfade zu AFP Fontressourcen und Ersatzschriften:	
AFPFontPath=<path>	-ap
Benannte, logische FTP Schnittstellen	
ResPath=<path>	-rp
ExtFontPath= <path>	-fp
Dateinamenserweiterungen für AFP Ressourcen:	
CharSetExt=<file extension>	
CodedFontExt=<file extension>	
CodePageExt=<file extension>	
Zeichenenkodierungen, basierend auf ASCII Code Pages	
CodePage= <defaultcodepage>	-dcp
CpPath=<path>	-cp
CodePageDefaultChar = " "	

Die Fontmapping Datei (mapping.def)

Die mapping.def steuert die Verarbeitung von Schrift während der Konvertierung.

AFP2web bietet eine Standardverarbeitung, in der immer versucht wird, die Originalschrift zu verwenden. Aus diesem Grund ist es prinzipiell immer möglich AFP2web ohne Änderung der mapping.def zu verwenden. Sie werden jedoch die mapping.def immer einsetzen, um die Schriftumsetzung für Ihre Dokumenttypen anzupassen..

Die Abschnitte der mapping.def:

- legen den Verarbeitungsmodus für jeden Character Set oder Schrift explizit fest
- legen die Zuordnung zu Ersatzschriften fest - für ein Zielformat wie AFP, PDF oder für ein Rasterformat wie TIFF separat für Windows oder für Unix Systeme.
- Ordnen Zeichensätze (Character Sets) zu Code Page Dateien.
- geben die möglichen Ergänzungen zu den Dateinamen an, damit AFP2web die Schriftdateien auffinden kann. Beispiel: Wenn Sie für die Schrift "ARIAL, Kursiv" eine Datei mit dem Namen Arial-Italic.ttf verwenden, so teilen Sie AFP2web mit, dass Ihre Schriftdateien im Namen den Suffix "-Italic" für eine Kursivschrift verwenden.
- beschreiben die Attribute der AFP Fontressourcen (Character Set, Code Pages, Schrift Global IDs). Einträge in diesen Abschnitten helfen, wenn der AFP Datenstrom beispielsweise auf nicht vorhandene Zeichensätze verweist oder wenn Sie die ursprünglichen AFP Fontressourcen nicht bereitstellen können. In diesem Falle definieren Sie AFP Fontressourcen manuell in den AFP-spezifischen Abschnitten der mapping.def.

Eine detaillierte Beschreibung von Syntax und Parametern finden Sie unter "*Parameter der mapping.def*" auf Seite 163.

Code Page Dateien

Code Page Dateien (CP-Dateien) definieren die Zeichencodes für Schrift. Sie bilden die EBCDIC Zeichencodes auf die entsprechenden ASCII und Unicode Zeichencodes ab. Die AFP2web Scripting Facility verwendet die ASCII Einkodierung, um die Objekte im AFP Datenstrom zu identifizieren.

AFP2web liefert einen Satz vordefinierter Code Page Dateien, die allgemein in Gebrauch sind und die in einem Verzeichnis zusammen mit der mapping.def abgelegt sind.

Weiteregehende Informationen finden Sie unter *"Code Page Dateien (CP-Dateien)"* auf Seite 179.

Standard Schriftzuordnung

Wenn weder der AFP Spool noch eine der AFP2web Konfigurationsdateien Informationen zu Fontressourcen liefern, richtet sich AFP2web nach Standardannahmen.

In diesem Fall kann AFP2web ein originalgetreues Konvertierungsergebnis nicht gewährleisten.

Beispiel: Schriftinformationen können von der IBM-spezifischen Namenskonvention für Ressourcen abgeleitet werden. Anhand dieser Merkmale ist AFP2web noch immer in der Lage, eine Ersatzschrift zu bestimmen. Eine Beschreibung dieser Regeln finden Sie unter *"Schriftzuordnung laut IBM Namenskonvention für Ressourcen" auf Seite 191*.

3.2 Die Verarbeitungsmodi für Schrift in AFP2web

AFP2web kennt die folgenden Verarbeitungsmodi für Schriften:

- Verwendung der Originalschrift (Standardverhalten von AFP2web)
- Schriftersetzung und -referenzierung
- Schriftersetzung und -einbettung

Das Standardverhalten von AFP2web

Normalerweise ist es nicht erforderlich AFP2web für die Verwendung von Schrift zu konfigurieren. Ist die Schrift inline oder als externe Ressource verfügbar, so verwendet AFP2web standardmäßig die Originalschrift für das Konvertierungsergebnis.

Die Standardkonvertierung von AFP Dokumenten

Allgemeine Hinweise

Bei der Konvertierung von AFP Dokumenten verwendet AFP2web die Originalschriften:

- Für die Ausgabe nach PDF konvertiert AFP2web die Schrift. Eine AFP Schrift vom Typ Outline Font wird zu einer Type 1 Schrift. Eine AFP Schrift vom Typ Raster Font wird zu einer Type 3 Schrift. In diesem Modus werden Schriften immer in das PDF eingebettet.
- Für die Ausgabe nach TIFF werden die Schriften gerastert.

Ist die AFP Schrift nicht inline im AFP Spool zu finden, so müssen diese als externe AFP Fontressourcen verfügbar sein.

Hinweis: *AFP Outline Fonts vom Typ "CID-Keyed" werden von AFP2web Version 3.x nicht unterstützt.*

Beispiel: AFP2PDF mit Charset Rendering=0 (Schrift rasterung)

Das Eingabeformat AFP soll zu PDF konvertiert werden. Die Originalschrift des AFP Character Sets C1H400B0 soll konvertiert und direkt in das PDF Dokument eingebettet werden.

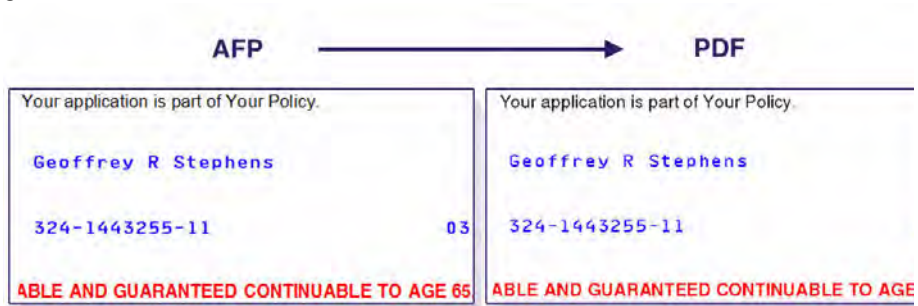
Die Konfigurationsdatei afpcp\mapping.def definiert, dass alle Schriften durch die Rasterung von Schrift umgesetzt werden sollen. Das entspricht dem Standardverhalten von AFP2web:

```
[CHARSET RENDERING]
*=0
```

Der folgende Aufruf von AFP2web startet die Konvertierung:

```
afp2web.exe -lf -lp:pdf -c -q samples\insure.afp
```

Wir erhalten ein PDF Dokument. Unser Beispiel bettet Schriften ein und ist daher größer 100 kB.



Die Informationen in der LOG-Datei:

```
==>CS=C1H400B0, FGI D=2305, TF=HELVETICA LATIN1, W=B, S=12.00, RF,
CP=T1GI 0361
-->UsedFont=C1H400B0, TF=F1, W=B, S=12.00, RF, ACP=MD: 2065. cp,
ENC=ANSI
  @(8496, 862)>Insurance<
  @(8496, 1120)>Specialists<
```

LOG-Text	Bedeutung
HELVETICA LATIN1	Der Name der Schrift ist laut AFP Definition "HELVETICA LATIN1"

LOG-Text	Bedeutung
C1H400B0	Die Ausgangsschrift in AFP. Der Name des AFP Character Sets. AFP2web verwendet die originale AFP Schrift für die Schriftrasterung.
F1	Die Schrift ist mit dem generierten Typeface-Namen F1 in das PDF Dokument eingebettet und dort als Type 3 Schrift erkennbar.
W=B	Die Schriftstärke ist "Fett"
RF	Rasterfont. Die Schrift wurde als Rasterschrift eingebettet.

Die Standardkonvertierung von PDF Dokumenten

Allgemeine Hinweise

Bei der Konvertierung von PDF Dokumenten zu AFP setzt AFP2web Schriften standardmäßig wie folgt um:

- Die PDF Schrift wird gerastert und als IOCA Images in das AFP Dokument eingefügt.
- Eine im PDF eingebettete Type 3 Schrift wird zu einer AFP Rasterschrift umgewandelt und als Ressource im Konvertierungsergebnis mitgeliefert.

Hinweis: Schriften mit einer Einkodierung nach Identity-H, Identity-V, Custom (Benutzerspezifisch) oder Built-In werden immer gerastert.

Beispiel: PDF2AFP mit CharSetRendering=0,1 (Standardeinstellung)

Das Eingabeformat PDF soll zu AFP konvertiert werden. Wir verlassen uns auf die Standardeinstellung von AFP2web.

Standardmäßig werden PDF Schriften für AFP gerastert und als Grafiken eingebettet. Es gibt keine AFP Fontressource als Konvertierungsergebnis. Dies entspricht der Einstellung in der mapping.def:

```
[CHARSET RENDERING]  
*=0,1
```

Der folgende Befehl in der demo.bat startet die Konvertierung:

```
afp2web.exe -q -c -afp -lf -lp:pdf samples\insure.pdf
```

Die Informationen in der LOG-Datei zeigen an, dass die eingebettete Schrift zu einer Rastergrafik umgesetzt wurde:

```
====>TF=Courier, W=M, S=10.00, OF,  
-->UsedFont=extfont/courier.ttf, TF=Courier, W=M, S=10.00, RI  
@(4896,10706)>Geoffrey R Stephens<
```

<i>LOG-Text</i>	<i>Description</i>
Courier	Die TrueType Schrift in der PDF Datei
UsedFont	Zeigt an, dass die Schrift zur Verarbeitung in ein temporäres Verzeichnis entpackt wird.
RI	Schrift ist als Raster Image eingefügt

Schriftersetzung und -referenzierung

Mögliche Gründe für Schriftersetzung und -referenzierung sind:

- Eine Ersatzschrift ist verfügbar und ihre Verwendung erzielt gute Konvertierungsergebnisse.
- Die Größe einer PDF Datei muss reduziert werden. Folglich muss die Schrift referenziert und nicht eingebettet werden.

Die Konvertierung von AFP mit Ersatzschriften

Allgemeine Hinweise

Sie verwenden die mapping.def um für AFP Schriften die Ersatzschriften festzulegen:

- Der Abschnitt **[CHARSET RENDERING]** definiert den Verarbeitungsmodus für jedes AFP Character Set.
- Für die Konvertierung nach PDF werden im Abschnitt **[PDFFONT]** AFP Character Sets ihren Ersatzschriften für PDF zugeordnet.
- Für die Konvertierung zu einem Rasterformat wie TIFF, JPEG, werden im jeweiligen Abschnitt für die Systemplattform **[WINFONT]** oder **[UNIXFONT]** die AFP Character Sets ihren Ersatzschriften zugeordnet.

Die AFP Schrift muss als Fontressource vorhanden sein - entweder inline im AFP Datenstrom oder als externe Ressource.

Weitere Angaben in der mapping.def bestimmen die Auswahl der Ersatzschrift:

1. Der AFP Datenstrom liefert den Namen der AFP Schrift.
2. Die mapping.def ordnet die AFP Schrift einem Typeface Namen zu.
3. Der mapping.def Abschnitt **[FONTSUFFIX]** legt die Namenskonvention für die Erweiterung des Dateinamens anhand der Schriftattribute (fett, kursiv) fest.
4. Die Schriftattribute der AFP Fontressource, ansonsten die Attribute aus dem mapping.def Abschnitt **[FGID]** liefern ergänzende Informationen.

AFP2web unterstützt die folgenden Formate als Ersatzschriften:

- TrueType,
- OpenType (nur als Single TrueType Font mit der Dateierweiterung ".ttf")
- Type 1

Beispiel: AFP2PDF mit CharSetRendering=1 (Schriftersetzung, -referenzierung)

Das Eingabeformat AFP soll zu PDF konvertiert werden.

Die Konfigurationsdatei afpcp\mapping.def definiert, dass alle Schriften durch entsprechende PC Schriften für PDF ersetzt und referenziert werden sollen. Ist eine PC Schrift im System für AFP2web nicht zu finden, so wird eine Standardschrift genommen. Wichtig ist, dass die PC Schriften für AFP2web im System verfügbar sind.

Die Einstellung in der mapping.def (ersetzen und referenzieren):

```
[CHARSET RENDERING]
*=1
```

Der folgende Aufruf von AFP2web startet die Konvertierung:

```
afp2web.exe -lf -lp:pdf -c -q samples\insure.afp
```

Wir erhalten ein PDF Dokument mit einer deutlich geringeren Dateigröße, da die Schriften referenziert werden. Unser Beispiel ist nur etwa 45 kB groß.

Die Informationen in der LOG-Datei zeigen hier, dass HELVETICA LATIN1 der PC Schrift Helvetica-Bold zugeordnet wird:

```
==>CS=C1H400B0, FGID=2305, TF=HELVETICA LATIN1, W=B, S=12.00, RF,
CP=T1GI0361
-->UsedFont=Helvetica-Bold, TF=Helvetica-Bold, W=B, S=12.00, SF,
ACP=MD: 2065. cp, ENC=ANSI
  @(8496, 862)>Insurance<
  @(8496, 1120)>Specialists<
```

<i>LOG-Text</i>	<i>Bedeutung</i>
C1H400B0	AFP Character Set mit dem Namen der Schriftfamilie HELVETICA LATIN1
UsedFont	Die verwendete PC Schrift ist Helvetica
SF	Standard Font

Die folgenden Zeilen in der LOG-Datei informieren darüber, dass AFP2web die PDF Schrift im System nicht finden kann. Folglich wird eine Warnung ausgegeben und stattdessen eine Standardschrift (erkennbar am Kürzel "D") verwendet:

```
==>CS=C1D0GT10 TF=GOTHIC Warning! External font file not found.
```

```
==>CS=C1D0GT10, FGI D=40, TF=G0THI C, W=M, S=10.00, RF, CP=T1D0BASE  
-->UsedFont=Helvetica, TF=Helvetica(D), W=M, S=10.00, SF,  
ACP=MD: 2063. cp, ENC=ANSI  
@(2448, 5353)>Geoffrey R Stephens<
```

<i>LOG-Text</i>	<i>Bedeutung</i>
Warning!	Für das AFP Character Set C1D0GT10 (Schriftname GOTHIC) der Warnhinweis, dass eine externe Schriftdatei nicht gefunden wurde.
UsedFont	Die verwendete PC Schrift ist Helvetica
D	Default (Standard) da eine Definition fehlt

Bei der Verwendung von Ersatzschriften ist es empfehlenswert, das Konvertierungsergebnis zu überprüfen. Die Zeichenfolge "Geoffrey R Stephens" im Original AFP verwendet, beispielsweise, eine nicht-proportionale Schrift:

Geoffrey R Stephens

Das Konvertierungsergebnis in PDF verwendet eine Proportionalschrift, da der mapping.def keine Zuordnung für den Typeface-Name GOTHIC definiert:

Geoffrey R Stephens

So sehen wir, dass eine Konvertierung nicht in jedem Fall optimal ist und dass wir an einigen Stellen Anpassungen vornehmen müssen.

Beispiel: AFP2PDF mit CharSetRendering=1 (Festlegung der Ersatzschrift)

Das Eingabeformat AFP soll zu PDF konvertiert werden.

Ist eine PDF Schrift im System für AFP2web nicht zu finden, so wird eine voreingestellte Standardschrift genommen. Das wollen wir jedoch verhindern.

Die folgende Einstellung in der Konfigurationsdatei afpcp\mapping.def gibt an, dass das AFP Character Set C1DOGT10 ersetzt und referenziert werden soll, alle anderen Schriften jedoch als Rasterschriften zu verwenden sind:

```
[CHARSET RENDERING]
C1DOGT10=1
*=0
```

An anderer Stelle wird die benutzerspezifische Ersatzschrift festgelegt. Der Dateiname ist **lucon.ttf** für die Schrift Lucida Console. Da es keine allgemeingültige Namenskonvention für Schriften gibt, richtet sich AFP2web nach dem Dateinamen. Wir geben daher die Dateinamen an und gehen davon aus, dass diese den Schriftnamen beinhaltet:

```
[PDFFONT]
;
; user defined
GOTHIC=lucon
```

Wir verwenden als Ablageort für die Schriftdatei **ExtFontPath=c:\WINDOWS\Fonts** und die Schriftdatei **lucon.ttf**.

Der folgende Aufruf von AFP2web startet die Konvertierung:

```
afp2web.exe -lf -lp:pdf -c -q samples\insure.afp
```

Die Informationen in der LOG-Datei zeigen, dass die Zuordnung zu der Ersatzschrift erfolgt ist. Das AFP Character Set C1DOGT10 ist mit dem Typeface "Lucida Console" ersetzt:

```
=>CS=C1DOGT10, FGID=40, TF=GOTHIC, W=M, S=10.00, RF, CP=T1D0BASE
-->UsedFont=C:/WINDOWS/Fonts/lucon.ttf, TF=LucidaConsole, W=M,
S=10.00, TT, Ref, ACP=MD:2063.cp, ENC=ANSI
@(2448,5353)>Geoffrey R Stephens<
```

LOG-Text	Bedeutung
TF=GOTHIC	Die AFP Schriftname ist "GOTHIC"
UsedFont	Die Schriftdatei, die als Ersatzschrift verwendet wird. Die Schriftname ist von dort abgeleitet: LucidaConsole
Ref	Weist darauf hin dass die Schrift im PDF referenziert wird

Das Konvertierungsergebnis für die Zeichenfolge "Geoffrey R Stephens" ist verbessert.
Das Original in AFP ist:

Geoffrey R Stephens

Das Konvertierungsergebnis in PDF ist:

Geoffrey R Stephens

Die Konvertierung von PDF mit Ersatzschriften

Allgemeine Hinweise

Sie definieren in der mapping.def, welche PC Schriften durch AFP Schriften zu ersetzen sind:

- Der Abschnitt **[CHARSET RENDERING]** bestimmt den Verarbeitungsmodus für jede PC Schrift.
- Für die Konvertierung zu AFP gilt: Der Abschnitt **[AFPFONT]** ordnet jede PC Schrift einem AFP Character Set und Code Page zu
- Für die Konvertierung zu einem Rasterformat wie TIFF, JPEG, werden im jeweiligen Abschnitt für die Systemplattform **[WINFONT]** oder **[UNIXFONT]** Ersatzschriften zugeordnet.

Für die Konvertierung von PDF zu AFP gilt:

- Eingebettete Schriften werden ohne Änderung übernommen. Referenzierte Schriften müssen als externe Schriften verfügbar sein. AFP2web sucht nach externen Schriften.
- Für die Suche nach einer Schrift, verwendet AFP2web den vollen Namen der Schrift. Beispiel: Für die Schrift in PDF mit dem Namen "Arial" sucht AFP2web nach einer Datei mit dem Namen "Arial.*" und findet gegebenenfalls "Arial.pfb" oder "Arial.ttf".
- Enthält der Name einer Schrift ein Sonderzeichen wie z.B. ', ' so muss dieses Zeichen auch im Dateinamen vorhanden sein. Beispiel: Ist der Schriftname "Arial,Bold" so sollte es eine Datei mit dem Name wie "Arial,Bold.ttf" geben.
- Für die Identifizierung einer in PDF als Teilmenge eingebetteten Schrift, ignoriert AFP2web die ersten 7 Zeichen des Schriftnamens. Beispiel: Ist in PDF eine Teilmenge der Schrift "NHAHHE+Arial" eingebettet, so sucht AFP2web nach dem Namen "Arial.*" und findet gegebenenfalls "Arial.pfb" oder "Arial.ttf".

Beispiel: PDF2AFP mit CharSetRendering=1

Das folgende Beispiel gibt an, dass eine Ersatzschrift für alle Schriften, deren Typeface-namen mit "Courier*" beginnt, verwendet werden soll:

```
[CHARSET RENDERING]  
Courier*=1
```

Ist der RenderingFlag 1 gesetzt, so müssen die Ersatzschriften für die Ausgabe nach AFP in der mapping.def angegeben sein. Hierzu ergänzen Sie, wenn notwendig, die Einträge im Abschnitt **[AFPFONT]**. Nachfolgend der Eintrag für die Schrift "Courier":

```
[AFPFONT]  
Courier=CZ4200, T1001141
```

Die verwendete AFP Ersatzschrift muss als Ressource (als Coded Font oder als AFP Character Set und Code Page) in dem vorgegebenen Ressourcenpfad zu finden sein. Den Ressourcenpfad definieren Sie in der INI-Datei als allgemeiner Ressourcenpfad (**ResPath**) oder als Pfad zu den AFP Fontressourcen (**AFPFontPath**).

Die folgenden Befehle in der demo.bat startet die Konvertierung:

```
afp2web.exe -q -c -afp -lf -lp: pdf samples\insure.pdf
```

Die folgenden Zeilen weisen darauf hin, dass die Schrift ersetzt wurde:

```
==>CS=CZ4200, TF=Courier, W=M, S=10.00, OF,  
-->CS=CZ4200, TF=Courier, W=M, S=10.00, OF, , ACP=PG-DEF: 500. cp,  
ENC=WinAnsiEncoding  
@(509,1115)>Geoffrey R Stephens<
```

<

LOG-Text	Beschreibung
CZ4200	Character Set im AFP Dokument
OF	Outline Font

Schriftersetzung und -einbettung

Mögliche Gründe für Schriftersetzung und -einbettung sind:

- Eine Ersatzschrift ist verfügbar und ihre Verwendung erzielt gute Konvertierungsergebnisse.
- Die Datei soll alle Ressourcen beinhalten, z.B. für die Archivierung.

Die Konvertierung von AFP zu PDF mit Ersatzschriften

Allgemeine Informationen über die Festlegung einer Ersatzschrift finden Sie unter ["Die Konvertierung von AFP mit Ersatzschriften" auf Seite 79.](#)

Beispiel: AFP2PDF mit CharSetRendering=2 (Zuordnung AFP GOTHIC zu Windows Gothic)

Das Eingabeformat AFP soll zu PDF konvertiert werden. AFP2web ordnet die AFP Schrift mit dem Typefacenamen GOTHIC automatisch einer Windows Schrift mit dem Dateinamen Gothic.ttf zu.

Die Konfigurationsdatei afpcp\mapping.def definiert, dass das AFP Character Set ersetzt und in PDF eingebettet werden soll. Alle andere Schriften sollen als Originalschrift gerastert werden:

```
[CHARSET RENDERING]
C1D0GT10=2
*=0
```

Wir verzichten darauf, eine benutzerspezifische Ersatzschrift festzulegen und gehen davon aus, dass AFP2web die Schriftdateien im System finden wird. Wir legen daher für PDF keine weitere Schriftzuordnungen fest:

```
[PDFFONT]
;
; user defined
```

Wir stellen stattdessen die Windows Schrift direkt zur Verfügung. Hierzu sorgen wir dafür,

- dass der Parameter der INI-Datei **ExtFontPath** genau auf den Pfad zur Schriftdatei weist und
- dass die Schriftdatei nach der **Namenskonvention** benannt ist (Typefacename=Dateiname.*).

Hinweis: Ist der Schriftgrad "Fett" ("Bold") und/oder der Schnitt "kursiv" ("Italic"), so bekommt der Dateiname einen weiteren Suffix. Beispiel: Sucht AFP2web nach der Schrift "GOTHIC,Bold", so verwendet AFP2web zuerst den Schriftnamen "Gothic" und fügt diesem den ersten Suffix hinzu, wie dieser im Abschnitt [FONTSUFFIX] der mapping.def definiert ist. In unserem Falle "Gothicb.*. Kann AFP2web keine Datei mit diesem Namen finden, so sucht es nach einer Datei mit dem zweiten Suffix, z.B. Gothic-Bold.*, und so weiter.

In der mapping.def definieren wir Suffixe für Dateinamen für die Schriftgrade "Fett" und "Kursiv" wie folgt:

```
[FONTSUFFIX]  
BoldSuffix=b, bd, -Bold  
ItalicSuffix=i, -Italic, -Oblique  
BoldItalicSuffix=bi, -BoldItalic, -BoldOblique
```

Unser Beispiel verwendet **ExtFontPath=c:\WINDOWS\Fonts** und die Schriftdatei **GOTHIC.ttf**.

Der folgende Aufruf von AFP2web startet die Konvertierung:

```
afp2web.exe -lf -lp:pdf -c -q samples\insure.afp
```

Die Informationen in der LOG-Datei:

```
==>CS=C1D0GT10, FGI D=40, TF=GOTHIC, W=M, S=10.00, RF, CP=T1D0BASE  
-->UsedFont=c:/WINDOWS/Fonts/GOTHIC.TTF, TF=CenturyGothic, W=M,  
S=10.00, TT, Emb, ACP=MD: 2063. cp, ENC=ANSI  
@(2448, 5353)>Geoffrey R Stephens<
```

LOG-Text	Bedeutung
TF=GOTHIC	Das AFP Character Set C1D0GT10 hat den Schriftnamen GOTHIC.
UsedFont	Die Schrift wird der Schriftdatei korrekt zugeordnet
Emb	Die Schrift wurde im PDF eingebettet

Die Zeichenfolge "Geoffrey R Stephens" als Original in AFP ist:

Geoffrey R Stephens

Das Konvertierungsergebnis in PDF ist:

Geoffrey R Stephens

Die Schriftersetzungs-Modi können nicht zu einer originalgetreuen Wiedergabe führen, wie Sie an diesem Beispiel sehen können. Wir stellen auch fest, dass wir das Ergebnis verbessern können, wenn wir stattdessen eine Nicht-Proportionalschrift als Ersatzschrift verwenden.

Wir verbessern das Ergebnis, wenn wir das AFP Character Set gezielt durch eine geeignete Nicht-Proportionalschrift ersetzen:.

```
[CHARSET RENDERING]
C1D0GT10=2
*=0
```

```
[PDFFONT]
;
; user defined
GOTHIC=COUR
```

Die Information in der LOG-Datei:

```
==>CS=C1D0GT10, FGID=40, TF=GOTHIC, W=M, S=10.00, RF, CP=T1D0BASE
-->UsedFont=c:/WINDOWS/Fonts/cour.ttf, TF=CourierNewPSMT, W=M,
S=10.00, TT, Emb, ACP=MD:2063.cp, ENC=ANSI
@(2448,5353)>Geoffrey R Stephens<
```

Die Zeichenfolge in PDF:

Geoffrey R Stephens

Kapitel 4: AFP2web Scripting Facility Benutzerhandbuch

Die AFP2web Scripting Facility ist eine Erweiterung zu AFP2web. Sie bietet eine Skripting Schnittstelle, mit der Sie das Standardverhalten von AFP2web anpassen. Dieses Benutzerhandbuch dient als Einführung in die Scripting Facility und beinhaltet:

- Grundlagen und Vorschläge für Anwendungsmöglichkeiten
- Einführung in die Parsingereignisse und in die Skripting-Funktionalität (Packages und Methoden)

Eine detaillierte Beschreibung der Syntax und Methoden der Skripting Schnittstelle finden Sie in der *"Scripting Facility API Reference"* auf Seite 213 und der *"Scripting Facility Schnell-Referenz"* auf Seite 197.

Skriptbeispiele sind in AFP2web mitgeliefert. Sie können diese Beispiele als Vorlagen für Ihre eigenen Skripte verwenden. Eine Beschreibung dieser Beispiele finden Sie im Anhang zu diesem Dokument.

4.1 Grundlagen der AFP2web Scripting Facility

Scripting Facility Anwendungen

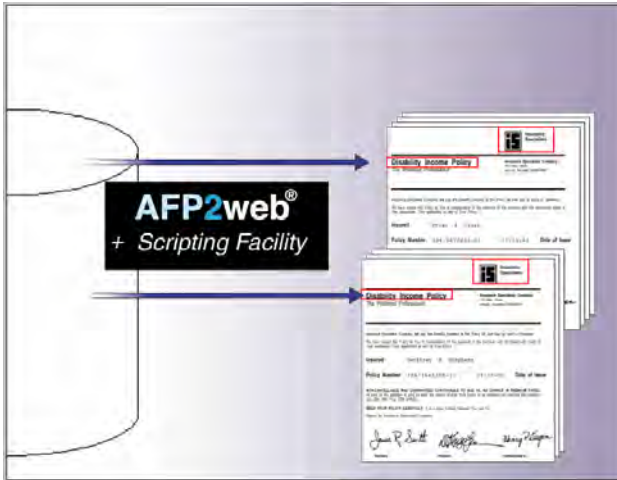
Die Scripting Facility ist die Schnittstelle zu AFP2web, mit der Sie das Standardverhalten von AFP2web anpassen. Wird die Scripting Facility aktiviert, übergibt AFP2web bei bestimmten Parsingereignissen die Kontrolle an ein benutzerdefiniertes Skript. Die Scripting Facility ermöglicht so eine intelligente Dokumentenverarbeitung. Beispiele für typische Anwendungen:

- Das Zerlegen einer AFP Spooldatei in einzelne Dokumente und Seiten nach den Regeln, die Sie definieren und die nicht von einer AFP Struktur fest vorgegeben sind. Enthalten Ihre AFP Daten die Standard AFP Indexelemente, die wir in diesem Kapitel beschreiben, so können Sie AFP2web diese Trennung automatisch ausführen lassen. Sie haben jedoch mit der Scripting Facility noch immer die Möglichkeit, die Dokumenttrennung nach zusätzlichen Anforderungen zu realisieren.
- Das Extrahieren von Daten aus dem Inhalt eines AFP Dokuments zur Erzeugung von Indexdaten. Auch in diesem Falle bietet AFP2web ein Standardverfahren für das Extrahieren von Daten aus den Standard AFP Indexelementen und für die Ausgabe dieser Daten in einem festgelegten Format. Verwenden Sie jedoch ein AFP Format in einer anwendungsspezifischen Variante oder wenn die AFP Indexelemente nicht die nötigen Informationen liefern, so bietet Ihnen die Scripting Facility die Möglichkeit, Indexdaten aus jedem beliebigen Objekt im AFP Datenstrom zu extrahieren und diese Daten auch in einem beliebigen Format auszugeben.
- Das Ändern, Hinzufügen, Entfernen von Dokumentinhalt wie Text, Wasserzeichen, PDF Lesezeichen, Hintergrundgrafiken, Grafiken, Indexdaten, Hyperlinkkommentare und andere Elemente. Wenn Sie den Dokumentinhalt bearbeiten, wieder verwenden oder ergänzen müssen, so verwenden Sie die Scripting Facility.
- Das Starten eines beliebigen vor- oder nachgelagerten Prozesses für die aktuelle Seite oder das aktuelle Dokument. Ein Beispiel hierfür: Sie veranlassen die Weiterleitung der ausgegebenen Indexdaten an ein Archivsystem. Statt es dem Archiv zu überlassen, regelmäßig nach eingehenden Daten zu suchen, veranlassen Sie die Aktualisierung des Archivs.
- Die Integration von AFP2web Funktionalität in eine bestehende Anwendung. AFP2web wird so Teil einer Prozesskette für die Dokumentverarbeitung. Zusammen mit der Scripting Facility produziert AFP2web in einem Schritt konvertierte Dokumente und Indexdaten für die nachgelagerten Prozesse und hilft so, den Ablauf zu optimieren.

Die Anwendungsmöglichkeiten der AFP2web Scripting Facility sind vielfältig. Wir beschreiben einige typische Anwendungsszenarios.

Dokumenterkennung und -trennung

Eine AFP Spooldatei in einzelne Dokumente und Dokumentseiten trennen.



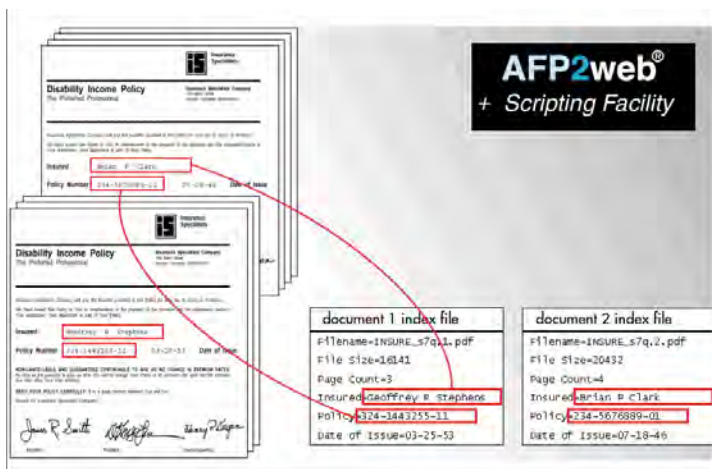
Ein Skript formuliert die Regeln für das Erkennen von Beginn und Ende eines Dokuments oder einer Dokumentenseite. Diese Regeln basieren auf das Vorhandensein von druckbarem und nicht-druckbarem Inhalt des AFP Spools wie gedruckter Text, Overlays oder andere Objekte, Werte in NOP-Feldern.

Die Dokumenterkennung und -trennung verläuft automatisch.

Beispiel: Ein AFP Datenstrom enthält einen Spool mit Versicherungspolicen für viele Kunden. Der Spool wird nach Kunden getrennt. Diese Trennung erfolgt nach der einfachen Regel: Die Startseite wird erkannt anhand eines Textes an einer fest vorgegebenen Position. Von dieser Seite extrahieren wir aus dem Text den Namen des Kunden für den Index.

Datenextraktion und -indizierung

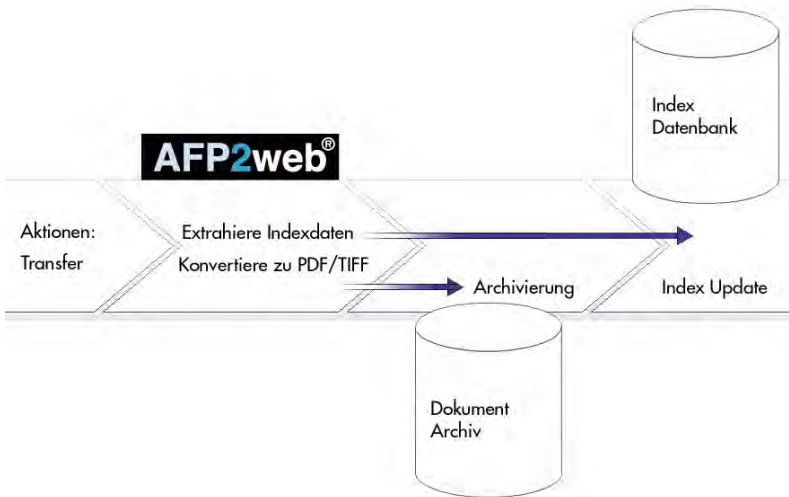
Extrahierung sichtbarer und nicht-sichtbarer Information aus AFP Dokumenten. Die Bereitstellung von Indexdaten in einem beliebigen Format, z.B. CSV oder XML.



Durch die Verwendung der Scripting Facility und eines maßgeschneiderten Skripts lässt sich sichtbarer Text und nicht-sichtbare Daten im Dokument erkennen. Indexdaten werden nach den Regeln im Skript extrahiert. Mit der Scripting Facility erfolgt die Datenextraktion von Massendruckdaten vollautomatisch. Die Indexdaten werden in jedem geforderten Format ausgegeben.

Dokumentkonvertierung und -archivierung

Verschiedene Aufgaben sind zu bewältigen: die Konvertierung eingehender Dokumente, das Herauslesen von Daten für den Index, die Bereitstellung der Dokumente für die Archivierung, die Ermittlung der technischen Dokument-ID des Archivs und die Bereitstellung von Indexdaten mit Dokument-ID für eine Aktualisierung der Indexdatenbank.

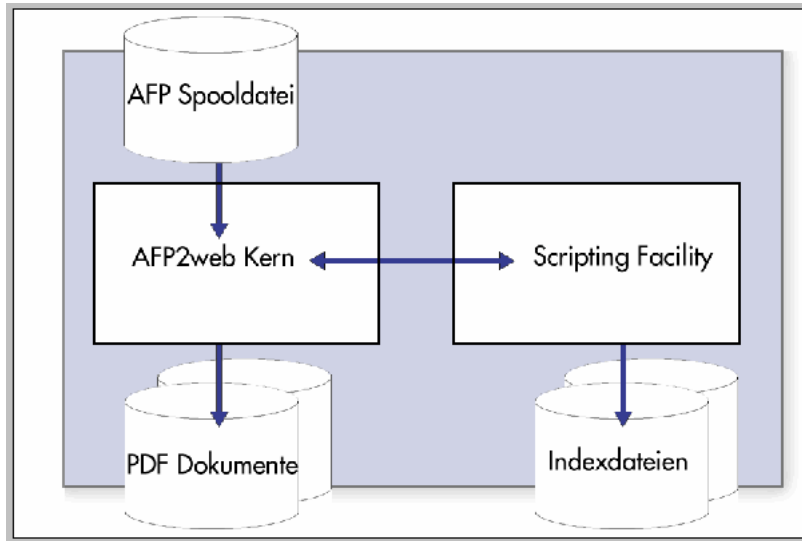


Dank der AFP2web Kernfunktionalität ist eine schnelle Konvertierung zu einem Archivformat wie PDF oder TIFF möglich. Mit der AFP2web Scripting Facility wird ein maßgeschneidertes Skript ausgeführt. Das Skript erstellt aus den druckbaren und nicht-druckbaren Dokumentdaten Indexdaten mit der erforderlichen Detailgenauigkeit und in jedem erforderlichen Format.

Nach der Dokumentkonvertierung fängt das Skript das "finalizeDoc()" Ereignis ab, um den nächsten Prozess für die Folgeverarbeitung anzustoßen. Zur Folgeverarbeitung gehören die Archivierung des Dokuments unter einer technischen Dokument-ID, die Weiterleitung der Dokument-ID mit den Indexdaten und die Aktualisierung der Indexdatenbank.

Scripting Facility Interaktion mit AFP2web

Mit der AFP2web Scripting Facility ist es möglich, bestimmte Ereignisse in AFP2web abzufangen, z.B. die Initialisierung, den Beginn oder das Ende eines Dokuments oder einer Dokumentenseite.



Die AFP2web Scripting Facility bietet eine Perl Engine (Perl DLL). Mit dieser können Sie Dokumente in AFP Spooldateien intelligent erkennen, trennen, konvertieren, indizieren und verteilen. Wenn Sie weitere Funktionalitäten oder neue Module hinzufügen möchten, müssen Sie Perl installieren.

AFP2web bietet Skriptbeispiele für einfache Aufgaben. Eine Beschreibung dieser Beispiele finden Sie im Anhang.

Die Objekte der AFP2web Scripting Facility

Für den Zugriff auf die Dokumente und deren Objekte verwenden Sie Perl Packages mit der Scripting Facility Funktionalität.

Die folgende Tabelle gibt einen Überblick der Scripting Facility Packages:

Scripting Facility Packages für den Zugriff auf Objekte in AFP2web

<i>Package</i>	<i>Ermöglicht den Zugriff auf</i>
a2w::Config	AFP2web Parameter
a2w::Document	Das Dokument
a2w::Font	Schrifteinstellungen
a2w::Index	Attribute für Indexdaten
a2w::Kernel	Bietet weitere AFP2web Funktionalität
a2w::Line	AFP Linie-Objekt
a2w::MediumMap	AFP Medium Map
a2w::NOP	Unsichtbare AFP "No Operation Fields" mit Daten oder Verarbeitungsanweisungen im AFP Datenstrom
a2w::Overlay	AFP Overlay
a2w::Page	Dokumentseite
a2w::PSEG	AFP Page Segment
a2w::Text	Text

Die Parsingereignisse der AFP2web Scripting Facility

Bei bestimmten Parsingereignissen wird eine Subroutine des Skripting-Hauptmoduls (standardmäßig afp2web.pm) gestartet. Die folgende Tabelle zeigt einen Überblick der AFP2web Ereignisse und die zugehörigen Subroutinen.

Parsingereignisse von AFP2web, Scripting Facility Subroutinen, Gültigkeitsbereich für Variablen (Scope)

<i>AFP2web Kernel</i>	<i>Subroutine in afp2web.pm</i>	<i>Scope</i>
Prozessbeginn	initialize() (einmal zu Beginn)	Global
Dokumentbeginn	initializeDoc()	Global

Die folgenden Subroutinen werden nacheinander pro Dokumentseite aufgerufen:

Seitenbeginn	initializePage()	Dokument
Seite ist geparst	afp2web()	Dokument
Seite ist verarbeitet	finalizePage()	Dokument

<i>AFP2web Kernel</i>	<i>Subroutine in afp2web.pm</i>	<i>Scope</i>
Dokument verarbeitet	finalizeDoc()	Global
Prozessende	finalize (einmal am Ende)	Global

Die erforderlichen Subroutinen in einem Skript

AFP2web übergibt beim Aufruf an jede Subroutine Objekte als Argumente. In der Subroutine sichern Sie eine Referenz auf ein Objekt, um nachher auf dessen Methoden zugreifen zu können. In der folgenden Beschreibung geben wir einige Hinweise zu den möglichen Aktionen in einer Subroutine. Syntaxbeschreibungen und Beispiele finden Sie in der englischsprachigen API Referenz, bzw. in den mitgelieferten Skriptbeispielen.

initialize()

Diese Routine wird einmal zu Beginn des AFP2web Prozesses ausgeführt. An dieser Stelle ermitteln Sie die Parameter (INI-Datei oder Kommandozeilenoption) und verändern diese notfalls während der Laufzeit. Auch können Sie Informationen über die Spooldatei abfragen.

Beispiele:

- Einlesen der Argumente, die als Kommandozeilenoption –sa übergeben werden
- Das Setzen der AutoSplit Eigenschaft auf TRUE aus dem Skript heraus, um die Dokumententrennung automatisch mit Hilfe der Standard AFP Indexelementen ausführen zu lassen

initializeDoc()

Diese Routine startet, wenn AFP2web beginnt, die Ausgabe eines Dokuments vorzubereiten. An dieser Stelle können Sie Prozesse initialisieren, die die resultierenden Dokumente weiterverarbeiten.

initializePage()

Diese Routine wird zu Beginn einer jeden neuen Dokumentseite gestartet. Von diesem Zeitpunkt an ist es möglich, auf alle Objekte einer Seite zuzugreifen.

afp2web()

Der Haupteinstieg für die AFP2web Scripting Facility.

Diese Routine wird für jede Dokumentseite ausgeführt. Möglich ist der Zugriff auf alle Objekte in der Seite, wie zum Beispiel:

- Referenzen auf alle untergeordnete Kinderobjekte sichern
- Objektmethoden für den Zugriff der Objekteigenschaften definieren
- Die Parsingregeln zur Erkennung der Startseite eines Dokuments, zum Überspringen von Seiten oder für das Hinzufügen der Seite zum aktuellen Dokument definieren
- Das Einsammeln der auf dieser Seite verfügbaren Indexdaten

finalizePage()

Diese Routine wird für jede Dokumentseite ausgeführt. Gestartet wird die Routine nach der Verarbeitung der Ausgabeseite. An dieser Stelle ist es möglich, Prozesse für die Fertigstellung der Seite anzustoßen.

finalizeDoc()

Diese Routine wird gestartet, nachdem das Dokument in die Ausgabe geschrieben wurde. Ab diesem Zeitpunkt ist die Dokumentkonvertierung zu PDF oder ein Rasterformat wie TIFF vollständig abgeschlossen.

Normalerweise verwenden Sie diese Routine, um Indexdaten zu formatieren und auszugeben. Sie können aber an dieser Stelle auch den Namen des Dokuments modifizieren, z.B. durch Erweiterung um einen Zeitstempel.

Da das ausgegebene Dokument von diesem Zeitpunkt an verfügbar ist, ist es auch möglich, jetzt die Nachbearbeitung wie z.B. die Archivierung zu starten.

finalize()

Diese Routine wird am Ende des AFP2web Prozesses nach der Verarbeitung des AFP Spools aufgerufen. Sie können weitere Folgeprozesse starten.

Scripting Facility Besonderheiten

Allgemeine Hinweise

Die Standard Zeichen-Encodierung (Default Encoding)

Beachten Sie bitte: Die Scripting Facility ist ASCII-basiert. Die ASCII Zeichentabellen liegen der Interpretation der AFP Ressourcennamen und AFP Indexelemente zugrunde. ASCII ist die standardmäßige Encodierung für die Scripting Facility.

Die mapping.def bestimmt, welche ASCII Zeichentabelle (ASCII Code Page) für ein AFP Code Page verwendet wird. Die ASCII Code Page Datei (CP-Datei) enthält die ASCII Codes für jedes Zeichen.

Das Koordinatensystem für Seitenmaße

Die Scripting Facility verwendet ein Koordinatensystem für die Dimension einer Dokumentenseite mit dem Ursprung links oben.

Die Zunahme des X-Werts erhöht die Dimension in die Richtung von links nach rechts.

Die Zunahme des Y-Werts erhöht die Dimension in die Richtung von oben nach unten.

Gültigkeitsbereich Global vs. Dokument (Zugriff auf Schriften)

Ein Skript arbeitet in zwei unterschiedlichen Kontexten, die sich auf den Gültigkeitsbereich von Einstellungen auswirken:

Global Scope beginnt mit der initialize() Routine und endet mit der finalize() Routine

Dokument Scope beginnt mit der initializeDoc() Routine und endet mit der finalizeDoc() Routine.

Die Einstellungen für eine Schrift, die während der Routine initialize() vorgenommen werden, gelten global.

Die Einstellungen für eine Schrift, die während einer der Routinen initializeDoc(), initializePage(), afp2web(), oder finalizePage() vorgenommen werden, gelten für das Dokument.

Wichtig: Es gibt eine Eigenheit von Perl: Wenn Sie eine Variable für eine Schrift deklarieren, vermeiden Sie bitte das Schlüsselwort "my", sonst steht die Variable global nicht zur Verfügung.

Die Verarbeitung von Indexdaten in einem Dokument

AFP2web unterscheidet zwei Arten von Indexdaten im AFP Datenstrom:

- Standard AFP Indizes
- Sonstige Indizes

Standard AFP Indizes

Elemente in einem AFP Datenstrom sind als Structured Fields kodiert. Für detaillierte Informationen über AFP und MO:DCA bitten wir, die entsprechende Dokumentation der IBM zu Rate zu ziehen. Im Folgenden beschreiben wir die Elemente, die wir die Standard AFP Indizes nennen.

Die Standard AFP Indizes sind Elemente im AFP Datenstrom vom Typ Index Element (IEL). Ein IEL befindet sich im Dokumentindex – innerhalb der Structured Fields Begin Document Index (BDI) und End Document Index (EDI).

Dem Index Element (IEL) folgt das Tag Logical Element (TLE). Das TLE liefert einen Indexeintrag, welcher aus Attributnamen und -wert besteht. Das TLE bezieht sich auf die Seite (Page) oder auf das Dokument (Page Group). Das TLE kann die Seite oder das Dokument explizit referenzieren. Ein Dokument oder eine Seite wird in diesem Zusammenhang generell auch als indiziertes Objekt bezeichnet.

Die Scripting Facility ist nicht unbedingt erforderlich, um diese Indexdaten auszugeben, denn AFP2web bietet bereits ein Verfahren für die Behandlung von Standard AFP Indizes. Sie aktivieren dieses Verfahren, wenn Sie in der INI-Datei den Parameter **FileCreationMode=DOC_INDEX** setzen. Mit dieser Einstellung erzeugen Sie Dateien mit Indexdaten in einem Standardformat. Weitere Informationen hierzu finden Sie in der allgemeinen Anleitung zu AFP2web.

Sie verwenden die Scripting Facility für Indexdaten vor allem dann, wenn Sie ein anderes Ausgabeformat für die Indexdaten benötigen oder wenn Sie die AFP Indizes neu erzeugen oder ergänzen wollen.

Sonstige Indizes

Sie verwenden möglicherweise eine unternehmensspezifische Lösung für die Indizierung. Häufig werden Indizes gewonnen aus:

- NOP (no operation) Structured Fields
- dem Erkennungstext (sog. "Eye Catchers") der Seite
- Text in Overlays

Mit der AFP2web Scripting Facility greifen Sie auf diese Elemente zu. Sie erzeugen neue Indizes oder erweitern vorhandene um weitere Informationen. Sie formatieren die Ausgabe nach Ihren Vorgaben.

Beispiel: Mit der `addIndex()` Funktion eines `a2w::Document` oder `a2w::Page` Objekts erzeugen Sie Indizes zu einem Dokument oder zu einer Seite. Werden diese Funktionen zusammen mit der Kommandozeilenoption `-bm` verwendet, entstehen aus den Indizes Lesezeichen für PDF.

Die Zerlegung eines AFP Spools in Dokumente und Seiten

AFP2web kann Dokumente automatisch trennen, wenn die AFP Daten Standard AFP Indizes enthalten.

Automatische Dokumententrennung (Autosplit)

Soll AFP2web die Dokumententrennung automatisch durchführen - bei gleichzeitigem Einsatz der Scripting Facility - müssen Sie in der Scripting Facility die **Autosplit** Funktion während der Initialisierung aktivieren.

Ist die automatische Dokumententrennung aktiv, startet AFP2web die Routinen für die Parsingereignisse automatisch. Sie können Indizes erzeugen und formatieren oder eine weitere Nachbearbeitung der Seite oder des Dokuments ausführen.

Sie haben in diesem Modus noch die Möglichkeit, die Dokumenttrennung zu beeinflussen: Sie können Schalter setzen, die bestimmen, dass eine Seite:

- dem aktuellen Dokument angefügt oder
- übersprungen werden soll.

Skriptkontrollierte Dokumententrennung ohne Autosplit

In vielen Fällen enthält die AFP Spooldatei nicht die Standard AFP Indizes, die AFP2web verwendet, um Dokumente zu erkennen. Möglicherweise müssen Sie Dokumente nach Ihren eigenen spezifischen Regeln trennen.

Wenn Sie die Dokumenterkennung ausschließlich in Ihrem Skript behandeln wollen, müssen Sie die Autosplit Funktion deaktivieren. Ihr Skript muss die Objekte im AFP Datenstrom untersuchen, zum Beispiel:

- Textobjekte in der Darstellung (sog. "Eye catchers")
- Overlays und deren Textobjekte
- NOP Felder

Ist die Autosplit Funktion ausgeschaltet, so ist Ihr Skript dafür verantwortlich, dass die Art der Dokumentseite erkannt wird. Sie müssen einen Schalter setzen, der signalisiert, dass

die aktuelle Seite:

- der Beginn eines neuen Dokuments ist.

Sie müssen auch Schalter setzen, die signalisieren, dass eine Seite:

- dem aktuellen Dokument angefügt oder
- übersprungen werden soll.

Wichtig: Ist die Autosplit Funktion ausgeschaltet, müssen Sie diese Schalter setzen. Nur so wird AFP2web die Parsingereignisse melden und die zugehörigen Skript-Routinen ausführen.

Fehlerbehandlung in einem Skript

Sie melden einen Fehlerfall aus einem Skript heraus, indem Sie einen negativen Rückgabewert und einen Meldetext an AFP2web zurückgeben. Die Syntax hierfür:

```
return (rc, "errortext").
```

wobei rc der negative Wert ist.

Erhält AFP2web einen negativen Rückgabewert, so beendet AFP2web die Verarbeitung mit einer Fehlerbenachrichtigung.

Beispiel:

```
return ( -123, "afp2web(): missing..." );
```

AFP2web gibt die Fehlerbenachrichtigung an die Konsole aus:

```
E088: Scripting Facility Error (rc=-123): afp2web(): missing...
```

4.2 AFP2web Scripting Facility verwenden

AFP2web liefert Skriptingbeispiele, die Sie als Ausgangspunkt für die Erstellung eines Skripts verwenden können.

afp2web.pm ist die Vorlage, die Sie verwenden sollten. In dieser Vorlage sind alle erforderlichen Subroutinen und auch Hinweise zu grundlegenden Funktionen zu finden.

Aktivierung der Scripting Facility

Sie aktivieren die AFP2web Scripting Facility mit Hilfe von Konfigurationseinstellungen in der INI-Datei oder indem Sie die entsprechenden Kommandozeilenoptionen beim Aufruf von AFP2web mitgeben. Eine Kommandozeilenoption ersetzt die entsprechende Einstellung in der INI-Datei.

Die nachfolgende Tabelle gibt einen Überblick über die INI Parameter und die entsprechenden Kommandozeilenoptionen:

<i>INI Parameter</i>	<i>Kommandozeilenoption</i>	<i>Beschreibung</i>
FileCreationMode=DOC_COLD	-doc_cold	<p>Option zur Aktivierung der AFP2web Scripting Facility.</p> <p>-doc_cold AFP2web gibt keine Indexdaten aus. Das ist Aufgabe des Skripts.</p> <p>Wenn Sie jedoch die Kommandozeilenoption -doc_cold erweitern, so gibt AFP2web die Standard AFP Indizes aus, bzw. die Indizes, die Sie mit der Methode addIndex() erzeugt haben. In diesem Fall übernimmt AFP2web die Formatierung und die Ausgabe der Indexdaten. Folgende Erweiterungen sind möglich:</p> <p>-doc_cold:csv AFP2web schreibt Indexdaten im CSV Format</p> <p>-doc_cold:xml AFP2web schreibt Indexdaten im XML Format</p> <p>Hinweis: Wenn Sie nur die Option doc_cold ohne Erweiterung eingeben, müssen Ihre Perl Routinen Indexdaten formatieren und ausgeben.</p>

<i>INI Parameter</i>	<i>Kommandozeilenoption</i>	<i>Beschreibung</i>
ScriptProcedure= <Skriptdatei>	-sp (Skriptdatei)	<p>Dateinamen des Skripts, das als Scripting-Hauptmodul geladen werden soll. Fehlt diese Angabe, so lädt AFP2web standardmäßig "afp2web.pm".</p> <p>Beispiel für die Kommandozeilenoption: -sp:c:\\afp2web\\mysp.pm</p> <p>Hinweis: Wird AFP2web nicht aus dem Verzeichnis der Installation aufgerufen, müssen Sie mit der Kommandozeilenoption -sp den absoluten Pfad zum Dateinamen mit angeben.</p>
ScriptArgument =parm1,parm2, ...	-sa:parm1,parm2, ...	<p>Mit dieser Option können Sie weitere Parameter an das Skript weiterreichen.</p> <p>Sie verwenden im Skript die Methode getScriptArgs() des a2w::Config Objekts, um auf diese Argumente zuzugreifen. Sie erhalten eine Zeichenkette, die Sie nach Ihren Regeln interpretieren (Trennungszeichen, Schlüsselwörter, etc.).</p>
Autosplit		<p>AFP2web sucht nach Standard AFP Indizes (TLEs) um Dokumentgrenzen zu erkennen. Ist Autosplit auf TRUE gesetzt, so hat der Schalter \$NEWDOC in einem Skript keinen Effekt.</p>
SkipPage, SkipObjectSize		<p>Dieser INI Parameter beschreibt Seiten, die übersprungen werden können. Als Kriterium legen Sie die maximale Bytegröße der Seite im Datenstrom fest. Alle Seiten kleiner oder gleich dieser Größe werden ignoriert. Leere Seiten aus einem Scan oder Trennblätter können übersprungen werden.</p>

Einstellungen für die Perl Umgebung

Die AFP2web Scripting Facility nutzt eine Perl Engine (wenn diese als Perl DLL mitgeliefert ist).

Wenn Sie AFP2web aus dem Verzeichnis der AFP2web Installation heraus aufrufen, benötigen Sie für die Basisfunktionalität der Perl Engine keine zusätzlichen Perl Module oder Perlumgebung.

Fall 1: Aufruf aus einem anderen Verzeichnis

Wenn Sie in Ihrem Skript zwar keine externen Module verwenden, jedoch AFP2web aus einem Verzeichnis außerhalb der AFP2web Installation aufrufen, dann müssen Sie Folgendes berücksichtigen:

- AFP2web mitteilen, wo die AFP2web Perl Packages (a2w*.pm) zu finden sind. Sie können hierzu die PERLLIB Umgebungsvariable setzen. Die einfachere Alternative ist es, im Skript das interne Perl Array @INC zu modifizieren, wie unter [*"Fall 2: Externe Module" auf Seite 106*](#) gezeigt.
- Der AFP2web Perl Engine mitteilen, welches Skript als Hauptmodul zu laden ist. (Standardmäßig ist dies afp2web.pm). Sie geben den Dateinamen des Skripts mit der Kommandozeilenoption -sp an.

Beispiel:

```
-sp: c:\AFP2web\MyModule.pm
```

Fall 2: Externe Module

Wenn Sie externe Module verwenden, müssen Sie der AFP2web Perl Engine mitteilen, wo diese zu finden sind. Sie können dies in der PERLLIB Umgebungsvariable definieren. Die einfachere Alternative ist es, im Skript das interne Perl Array @INC zu modifizieren.

Wichtig: Das Modifizieren von @INC muss die erste Anweisung sein in dem als Hauptmodul von AFP2web geladenen Skript.

Wir möchten anhand eines Beispiels zeigen, wie Sie zur Laufzeit weitere Verzeichnisse dem Perl Suchpfad hinzufügen. Die Annahmen sind:

- Ihr Skriptmodul (afp2web.pm) erfordert ein Perl Modul (XML::Writer), das sich in Ihrer Perl Installation in C:\Perl\ befindet.
- Ihr Skriptmodul befindet sich nicht im Verzeichnis der AFP2web Installation, sondern in C:\temp\xxx
- Die AFP2web Scripting Facility Packages (a2w:*) befinden sich nicht im Verzeichnis der AFP2web Installation, sondern in c:\temp.

Sie fügen diese neuen Verzeichnisse in den Perl Suchpfad mit der folgenden Anweisung in der ersten Zeile Ihres Skripts:

```
BEGIN {  
    unshift(@INC, ('C:\Perl\lib', 'C:\Perl\site\lib', 'C:\temp'));  
}
```

Sie geben dann beim Aufruf von AFP2web an, welches Skript geladen werden soll:

- entweder in der INI-Datei mit dem Parameter
ScriptProcedure=C:\temp\xxx\afp2web.pm
- oder mit der Kommandozeilenoption -sp:C:\temp\xxx\afp2web.pm.

Weitere Verwendungshinweise

Verwendung des INI Parameters MemoryOutputStream

Wenn AFP2web mit dem INI Parameter MemoryOutputStream=ON gestartet wird, so schreibt der AFP2web Kernel das Ausgabedokument in einen Speicherbereich und nicht in eine Datei.

Dieser INI Parameter ermöglicht den Zugriff auf das Ausgabedokument innerhalb eines Skripts. Hierfür stehen entsprechende Methoden des a2w::Document Packages (getOutputBuffer, getOutputBufferLength) zur Verfügung.

Ein Anwendungsfall ist die Nachbearbeitung eines Dokuments nach der Konvertierung (z.B. die Konvertierung, oder die Modifikation des Dokumentinhalts, etc.).

Der Zeitpunkt für den Zugriff auf das Ausgabedokument ist die finalizeDoc() Subroutine innerhalb eines Skripts.

Ein Beispiel für die Codierungsanweisungen für den Zugriff:

```
#---- Get PDF buffer length
$PDFBuffLenTmp = $a2wDocument->getOutputBufferLength();

#---- Get PDF buffer
$PDFBuffTmp = $a2wDocument->getOutputBuffer();

#---- From here on you can write the PDF buffer to a PIPE or
a database or a file or ...

#---- Sample to write to a file
my $OPFileNameTmp = "pdf/sf_" . $a2wDocument->getOutputFi-
lename();
open( OPFILE, ">$OPFileNameTmp" ) || die "Error! Unable to open
$0 : $!\n";
binmode( OPFILE );
syswrite( OPFILE, $PDFBuffTmp, $PDFBuffLenTmp );
close( OPFILE );
```

Kapitel 5: AFP2web Referenz

Die Referenz zu AFP2web enthält eine detaillierte Beschreibung der Parameter zur Steuerung der AFP2web Kernfunktionalität:

- INI Parameter (afp2web.ini)
- Kommandozeilenoption
- Parameter der mapping.def
- IBM-spezifische Konventionen für die Benennung von AFP Fontressourcen

Die Referenz zu AFP2web enthält die Scripting Facility Referenz:

- Überblick der Methoden und Script-Routinen
- Scripting Facility API Referenz (in Englisch)

Die Referenz zu AFP2web schließt mit einer Auflistung der:

- Fehlermeldungen und Codes

5.1 Gegenüberstellung: INI-Parameter, Kommandozeilenoptionen

Die folgenden Tabellen zeigen jeden Parameter der INI-Datei und - wenn vorhanden - deren entsprechende Kommandozeilenoption. Die Einträge sind nach Funktion gruppiert.

Überblick der INI-Parameter und Kommandozeilenoptionen

<i>Lizenzinformation</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
Licensee		Lizenznehmer
SerialNr		Seriennummer

<i>Eingabe-/Ausgabeformate</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
InputFormat	-<inputformat>	Eingabeformat
OutputFormat	-<outputformat>	Ausgabeformat

<i>Lesen von stdin/ Schreiben nach stdout</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
	-std	Beides: -si und -so
	-si	Lesen von stdin
	-so	Schreiben nach stdout
MemoryOutputStream		Schreibe konvertiertes Dokument in einen Speicherbereich statt in eine Datei.

<i>Art der Ausgabedateien (auch Bearbeitung von Index-Informationen)</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
FileCreationMode	-DOC / ALL / -DOC_INDEX / -DOC_COLD / -DOC_MERGE / -PAGE	Art der Ausgabedateien
PageOutput	-po	Eine Datei pro Seite des Ausgabedokuments erstellen (nur für TIFF als Ausgabeformat)

<i>Verarbeitung von Indexdaten</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
IndexFormat	-DOC_INDEX: CSV XML -DOC_COLD: XML CSV	Format für Indexdaten
IndexRecord		Weitere Felder für die Ausgabe von Indexdaten im CSV-Format.
PDFBookmark	-bm	PDF Lesezeichen für Indexelemente

<i>Generieren von Dateinamen und Ausgabeverzeichnis</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
FilenamePattern	-fnpt	Muster für die Benennung der Ausgabedateien
GenerateUniqueFile	-u	Eindeutige Namen für Ausgabedateien generieren
DocumentCount	-dc	Unterverzeichnisse anlegen und die Anzahl Dokumente pro Unterverzeichnis einschränken

Ordner (Verzeichnisse) festlegen		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
	-ip	Pfad zur afp2web.ini Datei (INI-Datei).
	-if	Pfad und Dateiname der INI-Datei
LogPath	-lp	Pfad für LOG-Output
ResPath	-rp	Standardpfad für AFP Ressourcen
CpPath	-cp	Pfad für ASCII Code Pages
OutputFilePath	-op	Pfad für Ausgabedokumente
ExtFontPath	-fp	Pfad zu weitere Type 1 und TrueType Schriften
IndexPath	-xp	Pfad für die Ausgabe von Indexdateien
OverlayPath	-ovp	Pfad zu AFP Overlay-Ressourcen
PageSegmentPath	-psp	Pfad zu AFP Page Segment-Ressourcen
FormDefPath	-fdp	Pfad zu AFP Formdef-Ressourcen
AFPFontPath	-ap	Pfad zu AFP Schrift-Ressourcen
TempPath	-tp	Verzeichnis für temporäre Arbeitsdateien. Standardmäßig gilt der vom System vorgegebene Pfad. Das Verzeichnis muss vorhanden sein und AFP2web muss die Lese-/Schreibberechtigung für dieses Verzeichnis haben.

Code Page Voreinstellungen		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
CodePageDefaultChar		Ersatzzeichen
CodePage	-dcp	Default AFP Code Page für die Umsetzung von Zeichenketten und Indizes aus der AFP Spooldatei nach ASCII

<i>Informationen zum Auffinden von AFP-Ressourcen</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
	-rf	AFP-Ressourcen
	-xf	Eingabedatei mit dem AFP Index zum AFP Dokument
	-fd	AFP FORMDEF
FormdefExt		Dateierweiterung für AFP FORMDEFs
OverlayExt		Dateierweiterung für AFP Overlays
PageSegExt		Dateierweiterung für AFP Page Segments
CharSetExt		Dateierweiterung für AFP Character Sets
CodePageExt		Dateierweiterung für AFP Code Pages
CodedFontExt		Dateierweiterung für AFP Coded Fonts

<i>Der Abschnitt [FORMDEF]</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
<formdefs und deren medium maps>		Liste von AFP Formdefs. Zu jedem AFP Formdef die Liste der Medium Maps

<i>LOG-Dateien, Meldungen, Statistiken</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
Quietmode	-q	Unterdrücke oder zeige Verarbeitungsmeldungen in der Systemkonsole
Logging	-l	LOG-Output
LoggingFont	-lf	LOG-Output: Schriftinformationen
LoggingLevel	-ll	LOG-Output gezielt auswählen

<i>LOG-Dateien, Meldungen, Statistiken</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
ExceptionLoggingLevel	-ell	Informationsdetails für Fehlermeldungen festlegen
Statistic	-s	Ausgabe einer Verarbeitungsstatistik (stat*.txt)

<i>Spoolfilebearbeitung</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
Strict	-Strict	Abbruch, wenn AFP-Ressourcen fehlen
Protocol	-p	Protokoll ausgeben mit Dok-IDs
StartingDocument	-sd	Dok-ID, für den Beginn der Konvertierung
EndingDocument	-ed	Dok-ID, für das Ende der Konvertierung
StartingPage	-pp:[fp][-tp]	Seitenauswahl (Startseite)
EndingPage	-pp:[fp][-tp]	Seitenauswahl (Letzte Seite)
SkipPage		Standardseite ignorieren
SkipObjectSize		Maximale Größe zur Erkennung der Standardseite

<i>Weitere Programm-Schalter</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
	-h oder -?	Hilfe zu AFP2web (Liste der Kommandozeilenoptionen)
	-v	Programmversion von AFP2web anzeigen
	-vall	Versionen aller AFP2web Komponenten anzeigen

<i>Weitere Programm-Schalter</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilen option</i>	<i>Funktion</i>
LaunchPreview	-pv	Betrachtungsprogramm starten und konvertiertes Dokument zeigen

<i>PDF Dokumentinformationen, Sicherheit, Viewereinstellungen, Konvertierungsparameter</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilen option</i>	<i>Funktion</i>
Creator		Autor
Title		Titel
Subject		Thema
Keywords		Schlagworte
PDFSecurity	-ps	PDF Sicherheit
PDFUIOptions	-pui	Betrachtereinstellungen
PDFWinOptions	-pwn	Dokumentdarstellung
PDFDocLimits		Speicherreservierung für PDF
Color	-c	Farbe

<i>TIFF (raster format) output</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilen option</i>	<i>Funktion</i>
FormatType	-TIFF: compressi- onformat oder: -compressionfor- mat	TIFF Format
Resolution	-pr	Auflösung
Color	-c	Farbe

<i>Konvertierung nach ASCII</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
MaxCols	-mc	Maximale Anzahl Spalten pro Zeile
MaxRows	-mr	Maximale Anzahl Reihen pro Seite
	-iASCII	Optimalwerte anzeigen für die Kommandozeilenoptionen -mc und -mr

<i>Konvertierung nach AFP</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
AFPComplianceLevel		AFP Compliance Level
CurveSamplingFactor	-csf	Faktor für die Rasterung von Kurven
PageRotation	-r	Seitenorientierung
PrintableLineWidth	-plw	Minimale Linienstärke für den Druck

<i>Optimierung von Eingebetteten Grafiken (Included Objects)</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
SaveOCDataOnDisk	-sod	Objektdaten auf Festplatte schreiben
TempPath	-tp	Verzeichnis für temporäre Arbeitsdateien. Standardmäßig gilt der vom System vorgegebene Pfad. Das Verzeichnis muss vorhanden sein.
JPEGQuality	-jq:n	Qualität der JPEG-Komprimierung, n zwischen 1 und 100

<i>Farbe, IOCA Grafiken nach PDF</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilenoption</i>	<i>Funktion</i>
CMYKTORGB	-nocmyktorgb	CMYK zu RGB konvertieren

<i>Farbe, IOCA Grafiken nach PDF</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilen option</i>	<i>Funktion</i>
Abschnitt [AFPColorTable]	-clr	RGB Werte für Standard AFP OCA Farben

<i>Dokumentanpassungen</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilen option</i>	<i>Funktion</i>
PageRotation	-r	Seitendrehung
Watermark	-wm	Wasserzeichen
OutputSize	-os	Seitengröße

<i>AFP2web Scripting Facility</i>		
<i>Parameter der INI-Datei</i>	<i>Kommandozeilen option</i>	<i>Funktion</i>
FileCreationMode=DOC_COLD	-doc_cold	Aktiviert die AFP2web Scripting Facility. Auch um das Format für die Ausgabe der Indexdaten festzulegen.
ScriptProcedure	-sp	Name der Skriptdatei
ScriptArgument	-sa	Argumente an die Skriptdatei
ScriptUnitBase		Maßeinheit
Autosplit		Dokumente automatisch trennen mit Hilfe der Standard AFP Indexelemente (nur für die Kommandozeilenoption -doc_cold)

5.2 Parameter der afp2web.ini

Dieser Abschnitt beschreibt die Parameter der INI-Datei in alphabetischer Reihenfolge. Eine Gruppierung nach Funktionen finden Sie unter *"Gegenüberstellung: INI-Parameter, Kommandozeilenoptionen"* auf Seite 111.

Funktion und Struktur

afp2web.ini ist eine Konfigurationsdatei, mit der Sie global die Parameter für AFP2web festlegen. afp2web.ini ist in die folgenden Abschnitten unterteilt:

- [Settings] für Konvertierungsparameter
- [FORMDEF] für die Zuordnung von FORMDEFs zu Medium Maps
- [AFPColorTable] für die Zuordnung von AFP OCA Farben zu RGB Werten
- [user-defined section] für den Zugriff auf entfernte AFP Ressourcen mittels FTP

Hinweise für die Verwendung:

Wenn Sie Einstellungen in der afp2web.ini verändern, beachten Sie bitte:

- Steht ein Semikolon (";") am Anfang einer Zeile, so gilt die Zeile als Kommentar.
- Ein Pfad ohne Laufwerksangabe wird als relativ zum aktuellen Verzeichnis interpretiert. Wir empfehlen: Bitte verwenden Sie möglichst absolute Pfadangaben!
- Sie müssen nicht auf eine bestimmte Groß-/Kleinschreibung achten.
- Die INI Datei erlaubt eine maximale Zeilenlänge von 1 K.

Abschnitt [Settings]

AFPComplianceLevel=<modcap>

Konformität des erzeugten AFPs mit einem der MO:DCA Austauschformate:

modcais2	MO:DCA Interchange Set 2
modcap	MO:DCA P (Standard)

Beispiel: AFPComplianceLevel=modcap

Standardwert: modcap

AFPFontPath=<Pfad>

Pfad zu AFP Schrift-Ressourcen

Es ist auch möglich, eine kommagetrennte Liste von Pfaden anzugeben.

Beispiel: samples/resource,c:\afp\fonts

Standardwert: samples/resource

Autosplit=<on/off>

Automatische Trennung in Dokumente und Seiten basierend auf Standard AFP Indexelementen. Nur für FileCreationMode=DOC_COLD.

Wenn Autosplit=on, so werden die Beginn/Ende-Parsingereignisse der Scripting Facility automatisch ausgelöst. Als Folge wird der Rückgabewert 2 der Routine `afp2web()` ignoriert.

Standardwert: off

CharSetExt=<Dateierweiterung>

Dateierweiterung für AFP Character Sets (Zeichensätze).

Standardwert: *

Hinweis: AFP2web sucht standardmäßig nach *<resource name>**. Aus diesem Grund ist es normalerweise nicht erforderlich, diesen Parameter zu verwenden. Wenn Sie jedoch die Kommandozeilenoption *-Strict* setzen und hier eine Dateierweiterung strikt vorgeben, dann sucht AFP2web nach einer Ressource mit der Dateiname *<resource name>.<Dateierweiterung>*.

CMYKTORGB=<on/off>

Nur für die Ausgabe von IOCA FS45 farbigen Bildern nach PDF.

on	Reduziere IOCA FS45 CMYK Farbebenen zu RGB.
off	IOCA FS45 CMYK Grafikfarbebenen weiterreichen.

Hinweis: CMYK für PDF resultiert in großen PDF Dateien und wird nur empfohlen für den Druck, nicht aber für die Online-Darstellung.

Standardwert: CMYKTORGB=on

CodedFontExt=<Dateierweiterung>

Dateierweiterung für AFP Coded Fonts

Standardwert: *

Hinweis: Normalerweise sucht AFP2web nach <ressourcename>*. Es ist daher nicht erforderlich diesen Parameter zu verwenden. Wenn Sie jedoch den Parameter -Strict verwenden und auch eine Dateierweiterung angeben, so sucht AFP2web nach <ressourcename>.<Dateierweiterung>.

CodePage=<defaultcodepage>

Definiert die Default Code Page für die Konvertierung von Ressourcennamen, NOPs und Indizes in der AFP Spooldatei zu ASCII. (Beispiele: Index Namen und Werte, SFI Namen wie Doc Name, Page Name, Ressourcename, NOPs.)

Die mapping.def ordnet diese AFP Code Page zu der Code Page Datei (CP file). Der Standardwert wird somit in der mapping.def gesetzt.

Standardwert: T1GIG500

CodePageDefaultChar=" "

Code Page Ersatzzeichen

Zu verwendendes Zeichen, falls ein Zeichen nicht zugeordnet werden kann.

Standardwert: " "

CodePageExt=<Dateierweiterung>

Dateierweiterung für AFP Code Pages

Standardwert: *

Hinweis: AFP2web sucht standardmäßig nach `<resource name> *`. Aus diesem Grund ist es normalerweise nicht erforderlich, diesen Parameter zu verwenden. Wenn Sie jedoch die Kommandozeilenoption `-Strict` setzen und hier eine Dateierweiterung strikt vorgeben, dann sucht AFP2web nach einer Ressource mit der Dateiname `<resource name>.<Dateierweiterung>`.

Color=<on/off>

Gilt für TIFF:UNCOMPRESSED, TIFF:JPEG, PDF und für JPEG als Ausgabeformate. Farbinformationen (Text und Bild) entweder farbig oder in Schwarz/Weiß-Darstellung ausgeben.:

off	Schwarz/Weiß (Standardwert)
on	Farbig

CpPath=<Pfad>

Pfad für Code Pages und die Datei `mapping.def`.

Standardwert: `afpcp`

Creator=<doc-info>

PDF Dokumenteigenschaft: Author (Autor)

Standardwert: Voreinstellung laut INI-Datei

CurveSamplingFactor=<Faktor>

Faktor für die Glättung von Kurven. Je niedriger der Wert, desto glatter die Kurven. Möglich ist ein Wert von 1 bis 65535. Gilt nur für die Konvertierung nach AFP.

Beispiel: `CurveSamplingFactor=64`

Standardwert: 64

DocumentCount=<n>

Erzeugt Unterverzeichnisse und schränkt die Ausgabe auf maximal `<n>` Dokumenten pro Unterverzeichnis ein.

Diese Funktion gilt nur zusammen mit der Einstellung `FileCreationMode=DOC_INDEX`.

Beispiel: Die Option DocumentCount=500 zusammen mit FileCreationMode=DOC_INDEX schränkt die Anzahl von Dateien auf 1000 Dateien pro Unterverzeichnis ein: Sie erhalten in jedem Unterverzeichnis jeweils 500 Ausgabedokumente und 500 Indexdateien.

EndingDocument=<n>

Das letzte zu konvertierende Dokument. Als <n> ist die Dokument ID anzugeben, wie diese aus dem Protokoll zu ermitteln ist.

Nur möglich, wenn DOC_INDEX oder DOC_COLD ausgewählt ist.

Standardwert: 0

EndingPage=<endpage>

Seitenauswahl (letzte Seite).

Standardwert: -1 (last page)

ExceptionLoggingLevel=<n>

Detaillierungsgrad für Fehlermeldungen festlegen.

Eine der folgenden Optionen ist möglich:

1	Nur die Fehlermeldung (Standardwert)
2	Dateiname, Zeilennummer, Fehlermeldung

ExtFontPath=<Pfad>

Pfad zu weiteren Type 1 und TrueType Schriften.

Standardwert: extfont

FileCreationMode=<option>

Art der Ausgabedateien:

ALL	Eine Ausgabedatei für alle AFP-Dokumente (Standardwert)
DOC	Mehrere Ausgabedateien: Jeweils eine Ausgabedatei für jedes AFP-Dokument
DOC_INDEX	Erstellt eine Ausgabedatei mit jeweils einer Indexdatei

DOC_COLD	Die AFP2web Scripting Facility aktivieren.
DOC_MERGE	Zu einer Ausgabedatei zusammenführen. Gilt für TIFF nach PDF und für AFP als Eingabeformat.
PAGE	Eine Datei pro Seite in einem AFP-Dokument

FilenamePattern=<pattern>,Keyword1,Keyword2...

Filename pattern (ohne Erweiterung) definiert das Muster für Dateinamen mit Hilfe von Zeichenfolgen und/oder Platzhalter für reservierte Keywords.

Die Syntax:

<pattern>,Keyword1,Keyword2...

<pattern> definiert das Formatierungsmuster und entspricht der Syntax eines C printf Befehls.

Soll ein Pattern einen TIMESTAMP Wert (siehe unten) formatieren, so verwenden Sie die Syntax eines C strftime Befehls. Die Keywords, die Sie angeben, bilden den Dateinamen entsprechend dem vorgegebenen Muster (Pattern). Der Datentyp eines Keywords muss mit dem Datentyp im Pattern übereinstimmen.

Mögliche Keywords:

DOCID	Dokument ID (Datentyp: Integer)
PAGEID	Seiten ID (Datentyp: Integer)
PID	Prozess ID (Datentyp: String)
TIMESTAMP[:<pattern>]	Aktueller Zeitstempel (Datentyp: String)
SPOOLNAME	Name der Spooldatei (Datentyp: String)

Standardwerte:

Das Standardformat für TIMESTAMP ist "%y%m%d%H%M%S"

Welcher Standardwert für FilenamePattern verwendet wird, hängt von den Einstellungen FileCreationMode und GenerateUniqueFile ab wie folgt:

<i>Bedingung</i>	<i>Standardwert FilenamePattern</i>
FileCreationMode ist entweder FILE_MERGE, DOC_INDEX, DOC_COLD, DOC, oder PAGE	"%s_%s.%d",SPOOLNAME,PID,DOCID
GenerateUniqueFile=on	"%s_%s.%d",SPOOLNAME,PID,DOCID

<i>Bedingung</i>	<i>Standardwert FilenamePattern</i>
FileCreationMode=ALL und GenerateUniqueFile=off	"%s",SPOOLNAME
PageOutput=on (bzw. bei Verwendung der Kommandozeilenoption -po)	"%s_%s.%ld.%ld",SPOOL-NAME,PID,DOCID,PAGEID

FormatType=<tiffoutputformat>

Gilt nur für TIFF: Ausgabeformat für TIFF festlegen. Gültig ist ein Wert aus:

G3	ITU Group III
G4	ITU Group IV (Standardwert)
JPG	TIFF mit JPEG Komprimierung
LZW	TIFF mit LZW Komprimierung
PACK	TIFF, gepackt
UNCOMPRESSED	TIFF unkomprimiert

FormdefExt=<Dateierweiterung>

Dateierweiterung für FORMDEF Ressourcen.

Standardwert: *

Hinweis: AFP2web sucht standardmäßig nach <resource name> *. Aus diesem Grund ist es normalerweise nicht erforderlich, diesen Parameter zu verwenden. Wenn Sie jedoch die Kommandozeilenoption -Strict setzen und hier eine Dateierweiterung strikt vorgeben, dann sucht AFP2web nach einer Ressource mit der Dateiname <resource name>.<Dateierweiterung>.

FormDefPath=<Pfad>

Pfad zu FORMDEF Ressourcen.

Es ist auch möglich, eine kommagetrennte Liste von Pfaden anzugeben.

Beispiel: samples/resource,c:\afp\formdef

Standardwert: samples/resource

GenerateUniqueFile=<on/off>

Eindeutige Dateinamen für Ausgabedateien erzeugen.

Dieser Parameter wird wegen der Rückwärtskompatibilität zu früheren Versionen weiterhin angeboten. Dieser Parameter ist nicht erforderlich für AFP2web ab der Version 3.x und wir raten von der Verwendung dieses Parameters ab.

on	Eindeutige Namen für Ausgabedateien generieren
off	Nicht erzeugen (Standardwert)

IndexFormat=<CSV/XML>

Format der auszugebenden Indexdatei. Gilt nur, wenn FileCreationMode=DOC_INDEX angegeben wurde.

CSV	Indexdatei im CSV Format
XML	Indexdatei im XML Format (Standardwert)

IndexPath=<Pfad>

Pfad für die Ausgabe von Indexdateien. Ist dieser Pfad nicht angegeben, schreibt AFP2web die Indexdateien in den Pfad für Ausgabedateien (OutputFilePath).

Standardwert: pdf

IndexRecord=<fieldlist>

Nur für IndexFormat=CSV.

Eine Liste von Schlüsselwörtern für Metainformationen, die zu jedem Indexeintrag hinzugefügt werden sollen, in der Reihenfolge, wie sie auszugeben sind. Jedes Schlüsselwort ist mit einem Trennungszeichen zu begrenzen (zum Beispiel mit einem Komma ",").

Gültige Schlüsselwörter für <fieldlist>:

PAGE_GROUP_NAME	AFP Group Name
PAGE_NAME	AFP Page Name
PAGE_COUNT	Anzahl Seiten
FILE_NAME	Name der Ausgabedatei
FILE_TYPE	ASCII/JPEG/PDF/TIFF
FILE_SIZE	Größe der Ausgabedateien
INDEX	Index als: <IndexName>=<indexValue>

Standardwert: IndexRecord=
PAGE_GROUP_NAME,PAGE_NAME,PAGE_COUNT,FILE_NAME,FILE_TYPE,INDEX

InputFormat=<input format>

Dateiformat für die Eingabe. Möglich ist ein Wert aus:

AFP	AFP Dokument (Standard)
TIFFIN	TIFF Dokument

JPEGQuality=<n>

Kompressionsqualität für Farbgrafiken.

Nur gültig für den Parameter SaveOCDataOnDisk=on.

Für n ist ein Wert zwischen 1 und 100 möglich:

1	Beste Kompression, geringste Qualität
100	Geringste Kompression, beste Qualität
50	ist der empfohlene Wert (Standardwert)

Beispiel: JPEGQuality=50

JPEGQuality wird verwendet, wenn eine Grafik in einem PDF Dokument als JPEG ausgegeben wird, zum Beispiel:

Wenn die Ausgabe von Schwarz/Weiß erfolgen soll, so werden farbige Grafiken in der Spooldatei zu JPEG Grafiken in Grautönen konvertiert.

Wenn die Ausgabe von Farbe erfolgen soll und ist "CMYKTORGB" auf "on" gesetzt, so werden IOCA FS45 CMYK JPEG Planes (Farbebenen) auf eine Ebene reduziert und als RGB JPEG im PDF ausgegeben.

Keywords=<doc-info>

Stichwörter im PDF Dokument.

Standardwert: Voreinstellung laut INI-Datei

LaunchPreview=<option>

Vorschau. Startet das geeignete Betrachterprogramm für die resultierende Ausgabedatei

(den Adobe Reader für PDF oder einen Betrachter für eine ASCII, TIFF oder JPEG Datei).

on	Ausgabedatei zeigen
off	Nicht zeigen (Standardwert)

Licensee=<name>

Der festgelegte Name des Lizenznehmers. Bitte Ihre Angaben in Anführungszeichen angeben. Beispiel: Licensee="Maas High Tech Software GmbH".

SerialNr und Licensee aktivieren AFP2web Funktionen für die lizenzierten Eingabe-/Ausgabeformate und Betriebssystem.

Standardwert: Voreinstellung laut INI-Datei

Logging=<on/off>

LOG-Datei (Teil 1: Parsing-Teil, Teil 2: Builder-Teil mit Schriftinformationen):

on	Erzeuge LOG-Datei. Entspricht LoggingLevel=1,2,3,4, FONT
off	Keine LOG-Datei (Standardwert)

Hinweis: Wenn *LoggingLevel=0* folgt, dann wird dieser Parameter deaktiviert. Wenn Sie das vermeiden wollen, stellen Sie *Logging* nach dem Parameter *LoggingLevel* in der INI Datei.

LoggingFont=<on/off>

LOG-Datei (Teil 2: Builder-Teil mit Schriftinformationen):

on	Erzeuge LOG-Datei, Teil 2 Entspricht LoggingLevel=FONT
off	Keine LOG-Datei (Standardwert)

Hinweis: Wenn *LoggingLevel=0* folgt, dann wird *LoggingFont* deaktiviert. Wenn Sie das vermeiden wollen, stellen Sie *LoggingFont* nach dem Parameter *LoggingLevel* in der INI Datei.

LoggingLevel=<n>

Syntax:

LoggingLevel=1,2,3,4,5,6,8,SF,[RES | (ResType)+]

Logging Level. Schränkt die Meldungen in der LOG-Datei ein. Möglich als Wert für <n> sind ein oder mehrere Werte aus:

0	Keine LOG-Datei (Standardwert)
1	Structured Fields (SFI)
2	Data Information der SFI
3	Repeating Groups
4	Patterns (wie Grafik)
5	FOCA Information
6	Mapping.def Zeilen am Ende der LOG-Datei einfügen. Diese Zeilen können als Vorlage für die Anpassung der mapping.def verwendet werden.
8	FTP Logging
ALL	Alle Information im FOCA
FONT	Builder Teil, Font-Information (entspricht LoggingFont=on)
Ressource Logging Levels	
RES	Alle Ressourcentypen
	oder eine der folgenden Ressourcentypen:
CDP	Code Pages
FDEF	Formdefs
FNTR	Fonts (Detaillierte Auflistung aller Schriften in der aktuellen Spooldatei)
IOB	Object Containers
MMAP	Medium Maps
OLY	Overlays
PSEG	Page Segments
Grafik Logging Levels	
INFO	Info-Meldungen aus der Grafikverarbeitung
WARN	Warn- und Info-Meldungen aus der Grafikverarbeitung

DEBUG	Warn-, Info- und Debugmeldungen aus der Grafikverarbeitung
--------------	--

Beispiel: `LoggingLevel=1,INFO`

Hinweis: Wenn *LoggingFont* folgt, dann gilt *LoggingFont*.

LogPath=<Pfad>

Pfad für die Ausgabe der LOG-Dateien.

Standardwert: log

MaxCols=<max columns>

MaxRows=<max rows>

Maximaler Wert für die Anzahl von Spalten und Zeilen auf einer Seite. Diese Parameter schränken die Größe einer Seite ein. Gilt nur für ASCII als Ausgabeformat.

Standardwert: Wenn kein Wert oder 0 angegeben ist, berechnet AFP2web die Werte für das jeweilige Dokument.

MemoryOutputStream=<on / off>

Weist den A2W Kernel an, das Ausgabedokument in einen Puffer des Arbeitsspeichers auszugeben statt in eine Datei. Nur gültig zusammen mit `FileCreationMode "DOC_COLD"`. Verwendungshinweise finden Sie in der Einführung und der API Referenz zur Scripting Facility.

Standardwert: Off (Dokument in eine Datei ausgeben).

OutputFilePath=<Pfad>

Pfad für die Ausgabedateien.

Standardwert: pdf

OutputFormat=<output format>

Ausgabeformat:

AFP	AFP Format
ASCII	ASCII Format mit Anpassungen in den Zeilen/Spalten-Positionierungen

JPEG	JPEG Format
PDF	PDF Format (Standardwert)
PNG	Portable Network Graphics
TIFF	TIFF Format

OutputSize=<Breite>,<Höhe>

Größe der resultierenden Ausgabedatei. Breite und Höhe der Seite werden in Pixeln angegeben. Ein Anwendungsbeispiel ist die Erzeugung von Miniaturansichten.

Standardwert: Das Dokument wird in der Originalgröße ausgegeben.

OverlayExt=<Dateierweiterung>

Dateierweiterung für Overlay Ressourcen.

Standardwert: *

Hinweis: AFP2web sucht standardmäßig nach <resource name> *. Aus diesem Grund ist es normalerweise nicht erforderlich, diesen Parameter zu verwenden. Wenn Sie jedoch die Kommandozeilenoption -Strict setzen und eine Dateierweiterung strikt vorgeben, dann sucht AFP2web nach einer Ressource mit dem Dateinamen <resource name>.<Dateierweiterung>.

OverlayPath=<Pfad>

Pfad zu Overlay Ressourcen.

Es ist auch möglich, eine kommagetrennte Liste von Pfaden anzugeben.

Beispiel: samples/resource,c:\afp\overlays

Standardwert: samples/resource

PageOutput=<on/off>

Ausgabe einer Datei pro Seite des Ausgabedokuments. Gilt nur für TIFF als Ausgabeformat.

Standardwert: off

PageRotation=<rotation>

Seitendrehung:

0	0 Grad im Uhrzeigersinn
90	90 Grad im Uhrzeigersinn
180	180 Grad im Uhrzeigersinn
270	270 Grad im Uhrzeigersinn
portrait	Hochformat
landscape	Querformat

Standardwert: 0

PageSegExt=<Dateierweiterung>

Dateierweiterung für Page Segment Ressourcen.

Standardwert: *

Hinweis: AFP2web sucht standardmäßig nach <resource name>*. Aus diesem Grund ist es normalerweise nicht erforderlich, diesen Parameter zu verwenden. Wenn Sie jedoch die Kommandozeilenoption -Strict setzen und hier eine Dateierweiterung strikt vorgeben, dann sucht AFP2web nach einer Ressource mit der Dateiname <resource name>.<Dateierweiterung>.

PageSegmentPath=<Pfad>

Pfad zu Page Segment Ressourcen.

Es ist auch möglich, eine kommagetrennte Liste von Pfaden anzugeben.

Beispiel: samples/resource;c:\afp\pagesegments

Standardwert: samples/resource

PDFBookmark=<option>

PDF Lesezeichen für Index Elemente erstellen:

on	Lesezeichen einfügen
off	Nicht einfügen (Standardwert)

PDFDocLimits=<limit>

Nur für PDF: Dieser Wert wird als Faktor für die Reservierung von Speicher wirksam (Schriften, Bilder, weitere PDF Objekte).

Erhöhen Sie diesen Wert, wenn Sie die Fehlermeldung erhalten: "Too many images in this PDF: nnn. Increase limits by mpdf_open()."

Die Berechnung für die Reservierung von Speicher:

```
Memory allocated(in bytes)
= 18200 * PDFDocLimits + 6892
Memory allocated(in Kbytes)
= (18200 * PDFDocLimits + 6892) / 1024
```

Die folgende Tabelle zeigt einige Beispiele für Einstellungen und die resultierende Speicherreservierung:

<i>PDFDocLimits</i>	<i>Speicher (Kbytes)</i>
20	362
200 (Standardwert)	3561
500	8893
1000	17780

PDFSecurity=[<Owner Password>][,<User Password>][,<Key Length>][,<Permission Flags>]]]

PDF Sicherheitseinstellungen.

Die Parameter sind fest vorgegeben und haben die folgende Bedeutung:

Owner Password	Optional Owner Password. Passwort für die Änderungen der Sicherheitseinstellungen
User Password	Optional User password, erforderlich für das Öffnen des PDF Dokuments.

Key Length	Optional. Verschlüsselungsstärke. Möglich ist ein Vielfaches von 8 zwischen 40 und 128 40 => Normale Verschlüsselung, 128 => Bessere Verschlüsselung Standard ist 40.
Permission Flags	Optional. Eine oder mehrere Schalter für Berechtigungen, jeweils getrennt mit " ". Als Schalter ist ein Wert aus der folgenden Liste zu nehmen. Wenn Key Length 40, so sind die Schalter 5,6,7,8 nicht erlaubt.

Die folgenden Werte sind als Permission Flags möglich:

0	Keine Berechtigungen (Standardwert)
1	Drucken in niedriger Auflösung zulässig
2	Dokumentänderungen zulässig
3	Dokumentinhalt kopieren oder extrahieren zulässig
4	Kommentare eingeben und das Ausfüllen von Formularfeldern zulässig
5	Eingabe in Formularfelder oder von Signaturen zulässig
6	Inhalt für Ausgabehilfe entnehmen zulässig
7	Dokumentzusammenstellung zulässig
8	Drucken in hoher Auflösung zulässig

Beispiele:

1. Weder Owner noch User Password (Standardwert)

PDFSecurity=

2. Owner Password ist MHT, Verschlüsselungslänge ist 128 Bit, keine weitere Berechtigungen

PDFSecurity=MHT,,128,0

PDFUIOptions=[<HideMenubar>][,<HideToolbar>][,<HideWindowUI>]

PDF Betrachtereinstellungen ("User Interface")

Für jeden Positionsparameter, geben Sie entweder "on" oder "off" (ohne die Anführungszeichen) ein.

Der Standardwert ist "off" für jeden Parameter.

HideMenubar	on = Menüleiste wegblenden
HideToolbar	on = Werkzeugleiste wegblenden
HideWindowUI	on = Fensterkontrollen wegblenden

Hinweis: Wird ein Wert weggelassen, so gilt der Standardwert. Beispiel: PDFUIOptions=,on,on entspricht PDFUIOptions=off,on,on.

Beispiele:

Nur die Menüleiste wegblenden: PDFUIOptions=on

Menüleiste und Fensterkontrollen wegblenden: PDFUIOptions=on,,on

PDFWinOptions=[FitWindow][,<CenterWindow>][,<FullScreenWindow>][,<DisplayDocTitle>]

PDF Betrachtereinstellungen für die Darstellung im Fenster.

Für jeden Positionsparameter, geben Sie entweder "on" oder "off" (ohne die Anführungszeichen) ein.

Der Standardwert ist "off" für jeden Parameter:.

FitWindow	on = Dokumentgröße in das Fenster einpassen
CenterWindow	on = Dokument im Fenster zentrieren
FullScreenWindow	on = Dokument im Vollbildschirmmodus zeigen, ohne Menüleiste, Fensterkontrollen oder sonstige Fenster
DisplayDocTitle	on = Die Dokumenteigenschaft Title in der Fenstertitleiste zeigen off = Die PDF Dateiname in der Fenstertitleiste zeigen

Hinweis: Ein leerer Wert ist erlaubt und in diesem Falle gilt der Standardwert. Beispiel: PDFWinOptions=,on,on,on entspricht PDFWinOptions=off,on,on,on.

Weitere Beispiele:

Fenstergröße ändern: PDFWinOptions=on

Dokument zentrieren und im Vollbildschirmmodus zeigen: PDFWinOptions=,on,on

Dokumenttitel zeigen: PDFWinOptions=,,,on

PrintableLineWidth=<n>

Minimale Linienstärke, um zu gewährleisten, dass Linien ausgedruckt werden.
Anzugeben in Pixeln. Gilt für die Konvertierung nach AFP.

Beispiel: PrintableLineWidth=1

Standardwert: 1

Protocol=<on/off>

Verarbeitungsprotokoll mit Dokument IDs ausgeben. Diese Option ist nur möglich mit DOC_INDEX oder DOC_COLD.

on	Protokoll ausgeben
off	nicht ausgeben (Standardwert)

Quietmode=<on/off>

Meldungen an die Systemkonsole unterdrücken:

on	Keine Meldungen
off	Meldungen ausgeben (Standardwert)

Resolution=<dpi resolution>

Gilt nur für Rasterformate: Auflösung in dpi. Üblich ist ein Wert aus:

200	
240	Standardwert
300	

ResPath=<Pfad>

Pfad zu AFP Ressourcen.

Standardwert: samples/resource

SaveOCDataOnDisk=<on/off>

Objektdaten auf Festplatte schreiben.

Legt fest, ob Objektdaten temporär für die Verarbeitung im Pfad laut "TempPath" abgelegt werden oder nicht. Diese Option ermöglicht es, Arbeitsspeicher einzusparen und die Ausgabegröße der Objekte zu optimieren.

Standardwert: off

ScriptArgument=<Argumente>

Für die AFP2web Scripting Facility

(nur wenn FileCreationMode=DOC_COLD).

Argumente, die an das Skript zu übergeben sind.

Standardwert: Leer/Null (keine Argumente)

ScriptProcedure=<Scriptdatei>

Für die AFP2web Scripting Facility

(nur wenn FileCreationMode=DOC_COLD).

Dateiname mit der Skriptprozedur.

Standardwert: afp2web.pm

ScriptUnitBase=<Basisseinheit>

Basis Maßeinheit für ein Dokument. Diese Maßeinheit wird von der Scripting Facility zur Interpretation von Positions- und Größenangaben verwendet.

(Gilt nur für FileCreationMode=DOC_COLD).

Möglich ist ein Wert aus:

pixel	Pixel gemäß der Seitenauflösung (Standardwert)
mm	Millimeter

SerialNr=<seriennr>

Die zugeteilte Seriennummer für das Produkt AFP2web.

SerialNr und Licensee aktivieren die AFP2web Funktionalität für die lizenzierten Optionen für Eingabe-/Ausgabeformate und Betriebssystem.

SkipPage=<on/off>

Leere Seiten überspringen (Beispiel: eine eingescannte Leerseite). Mit "on" weist dieser Schalter AFP2web an, eine Seite zu ignorieren, wenn diese leer ist, bzw. wenn diese eine Grafik enthält, das in der Datenmenge die Angabe in "SkipObjectSize" nicht überschreitet.

Standardwert: off

SkipObjectSize=<n>

Die maximale Größe in Bytes für eine Grafik, die aus dem Scan einer leeren Seite resultiert. Eine leere Seite kann ignoriert werden. Beispiel: Eine eingescannte leere Rückseite eines Dokuments führt zu einer Grafik mit wenig Pixel. Mit diesem Parameter legen Sie die maximale Größe einer solchen Grafik fest.

Standardwert: 2K (=2048 Bytes)

StartingDocument=<n>

Das erste zu konvertierende Dokument. Als <n> ist die Dokument ID anzugeben, wie diese aus dem Protokoll zu ermitteln ist.

Nur möglich, wenn DOC_INDEX oder DOC_COLD ausgewählt ist.

Standardwert: 0

StartingPage=<startpage>

Seitenauswahl (Startseite)

Standardwert: 1

Statistic=<on/off>

Ausgabe einer Verarbeitungsstatistik:

on	Statistik ausgeben (Standardwert)
off	Statistik nicht ausgeben

Strict=<on/off>

Konvertierung abbrechen, wenn eine erforderliche AFP Ressource nicht gefunden wer-

den kann.

on	Abbrechen, wenn Ressource nicht vorhanden
off	Verarbeitung fortsetzen (Standardwert)

Subject=<doc-info>

PDF Dokumenteigenschaft: Subject (Thema)

Standardwert: Voreinstellung laut INI-Datei

TempPath=<Pfad>

Pfad für temporäre Dateien. Der Pfad muss vorhanden sein.

AFP2web muss die Lese-/Schreibberechtigung für dieses Verzeichnis haben. Wenn nicht, so bricht AFP2web beim Versuch ab, temporäre Dateien zu schreiben oder zu löschen.

Standardwert: Die Voreinstellung laut Betriebssystem.

Title=<doc-info>

PDF Dokumenteigenschaft: Titel

Standardwert: Voreinstellung laut INI-Datei

Watermark=<text, typeface,height,rotation,redmask,greenmask,bluemask>

Stellt Text als Wasserzeichen hinter dem Dokumenttext dar. Die möglichen Parameter:

text	Text
typeface	Schrift Typeface
height	Schriftgröße in Zehntel Punkt. Geben Sie 90 für 9 Punkt.
rotation	Rotationswinkel. Ein Wert von 0 bis 360.
redmask	RGB Red Wert von 0 bis 255
greenmask	RGB Green Wert von 0 bis 255
bluemask	RGB Blue Wert von 0 bis 255

Beispiel: Watermark=Confidential,Helvetica,600,60,189,219,231

Standardwert: kein Wasserzeichen

Parameter des Abschnitts [FORMDEF]

<formdef>= <medium map-list>

Dieser Abschnitt ist erforderlich, wenn die AFP2web Spool ein Formdef erfordert und diese nicht mit der Kommandozeilenoption -fd oder -rf spezifiziert wurde.

Der [FORMDEF] Abschnitt der INI-Datei ordnet Medium Maps zu den Namen von Formdef Ressourcen. Diese Definition ist hilfreich, wenn beispielsweise die Kommandozeilenoption -fd nicht verwendet wird. Die Formdef muss als externe Ressource verfügbar sein.

In diesem Abschnitt wird zu jeder Formdef die Medium Maps aufgelistet. Ein Medium Map sollte eindeutig einer Formdef zugeordnet sein. AFP2web lädt die erste Formdef, die es für ein gegebenes Medium Map finden kann.

Beispiel:

F1hqsd=HOCH, HOCHD, QUER, QUERD, HFACH1

Parameter des Abschnitts [AFPColorTable]

Der Abschnitt [AFPColorTable] legt die RGB Werte für jede Farbe in der AFP Standard OCA Farbtabelle fest. Wenn Sie die Standardwerte nicht wünschen, so ändern Sie diese Einstellung in diesem Abschnitt. (Detaillierte Informationen über die Standardwerte finden Sie in der Dokumentation der IBM, Document Number SC31-6802-06, Seite 473/501).

Ein RGB Wert liegt zwischen 0 und 255. Werte außerhalb dieses Intervalls werden als Wert 0 behandelt.

Syntax:

Farbname=<Red>, <Green>, <Blue>

Beispiel:

Blue=0, 0, 255

Farbnamen und die RGB Standardwerte:

Black	0,0,0
Blue	0,0,255
Brown	144,48,0
Cyan	0,255,255
Darkblue	0,0,170
Darkcyan	0,146,170
Darkgreen	0,146,0
Default	0,0,0
Gray	131,131,131
Green	0,255,0
Magenta	255,0,255
Medium	255,255,255
Mustard	196,160,32
Orange	255,128,0
Purple	170,0,170
Red	255,0,0
White	255,255,255
Yellow	255,255,0

Frei definierbare Abschnitte für Logische Schnittstellen

Sie können eine beliebige Anzahl frei benennbarer Abschnitte einfügen, um die Quellen für externe AFP Ressourcen zu definieren. Die möglichen Parameter:

[mysectionname]	Bitte die eckigen Klammer "["] verwenden, um eine FTP Verbindung zu definieren. Der Name, den Sie als mysectionname eingeben ist frei wählbar (Beispiel: [FONT300]). Wenn Sie die FTP Adresse referenzieren wollen, dann verwenden Sie diesen Namen mit dem Präfix @ in den jeweiligen INI Parametern. Beispiel: AFPFontPath=@FONT300
Type=FTP	Kommunikationsprotokoll
HostName=<hostname oder IP address>	Host Name oder IP Adresse
UserName=<username>	FTP Benutzername
Password=<password>	Passwort
StartDir=<starting directory>	Ausgangsverzeichnis Beispiele: StartDir='SYS1.FONT300' ==> Mainframe PO Data Set (Der Wert muss in einfachen Anführungszeichen stehen) StartDir=c:/afp/font3000 ==> Ein Windows FTP Server StartDir=/opt/afp/font3000 ==> Ein Unix FTP Server

Sie definieren jeweils ein Abschnitt pro FTP Adresse. Beispiel:

```
[MYFONT300]
Type=FTP
HostName=192. 168. 1. 1
UserName=afp2web
Password=*****
StartDir=' SYS1. FONT300'
```


5.3 AFP2web Kommandozeilenoptionen

Die folgende Beschreibung listet die Kommandozeilenoptionen in alphabetischer Reihenfolge auf.

Verwendungshinweise

Sie beginnen jede Kommandozeilenoption mit dem Zeichen '-'.

Wenn Sie keine Kommandozeilenoption eingeben, so wird entweder der Parameter der INI-Datei oder der Standardwert für diesen Parameter wirksam. Für die Priorität gilt die folgende Reihenfolge:

- Kommandozeilenoption beim Aufruf von AFP2web
- Parameter in der INI-Datei
- Standardwert (Siehe die Beschreibung der INI-Datei)

Anführungszeichen um mehrere Argumente in einer Option garantieren, dass alle Argumente weitergereicht werden. So kann verhindert werden, dass AFP2web die Argumentenkette an dem ersten Leerzeichen abbricht.

Beispiel:

-wm:"Confidential Text,Helvetica,600,60,189,219,231"

-<filecreationmode>

Art und Anzahl der Ausgabedateien. Als Wert für <filecreationmode> möglich ist ein Wert aus:

ALL	Eine Ausgabedatei für alle AFP-Dokumente
DOC	Jeweils eine Ausgabedatei für jedes AFP-Dokument
DOC_INDEX	Ausgabedateien mit Indexdateien (Siehe die Beschreibung weiter unten.)
DOC_COLD	Die AFP2web Scripting Facility aktivieren. Indexdateien mit der Scripting Facility AutoSplit Funktion erzeugen. (Siehe die Beschreibung unten.)
DOC_MERGE	Zu einer Ausgabedatei zusammenführen. Gilt für TIFF nach PDF und für AFP als Eingabeformat.
PAGE	Eine Datei pro Seite in einem AFP-Dokument

Das Format für die zu erzeugende Indexdatei wird festgelegt durch Erweiterung der Option:

-DOC_INDEX:CSV	Indexdatei im CSV Format
-DOC_INDEX:XML	Indexdatei im XML Format

Zusammen mit der AutoSplit Funktion der AFP2web Scripting Facility, gibt DOC_COLD mit einer Erweiterung an, wie Indexdaten zu verarbeiten sind bzw. auch das Format für die resultierenden Indexdateien:

-DOC_COLD	AFP2web erstellt keine Indexdateien. Diese Aufgabe wird von dem angepassten Skriptmodul übernommen.
-DOC_COLD:CSV	AFP2web erstellt die Indexdatei im CSV Format.
-DOC_COLD:XML	AFP2web erstellt die Indexdatei im XML Format.

-<input format>

Dateiformat für die Eingabe. Standardmäßig gilt AFP als Eingabeformat. Als Kommandozeilenoption kann ein hiervon abweichendes Format festgelegt werden:

TIFFIN	TIFF Dokument
---------------	---------------

-<output format>

Dateiformat für die Ausgabe

AFP	AFP als Ausgabeformat
ASCII	ASCII-Format mit Anpassungen in den Zeilen/Spalten-Positionierungen
JPEG	JPEG Format
PDF	PDF Format
PNG	Portable Network Graphics
TIFF	TIFF Format
TIF	TIFF Format

Ein TIFF-Subformat wird festgelegt durch Erweiterung der Option, wie folgt

-TIFF:G3

-TIFF:G4
-TIFF:JPG
-TIFF:LZW
-TIFF:PACK
-TIFF:UNCOMPRESSED

-<tiffoutputformat>

Das spezifische TIFF Format kann als weitere Kommandozeilenoption nach dem Parameter -TIFF mit angegeben werden. Möglich ist ein Wert aus:

-G3
-G4
-JPG
-LZW
-PACK
-UNCOMPRESSED

-ap:<pfad>

Pfad zu AFP Fontressourcen

Es ist auch möglich, eine kommagetrennte Liste von Pfaden anzugeben.

Beispiel: -ap:samples/resource,c:\afp\fonts

-bm

PDF Lesezeichen für Indexelemente erstellen.

-C

Farbe ausgeben.

Gilt für die Ausgabeformate JPEG, PDF, PNG, TIFF:JPEG und TIFF:UNCOMPRESSED.

-clr:<colorname>,<r>,<g>,

RGB Wert für die genannte Farbe. (Ersetzt die Definition im Abschnitt [AFPColorTable]) der INI-Datei. (Detaillierte Informationen über die Standardwerte finden Sie in der Dokumentation der IBM, Document Number SC31-6802-06, Seite 473/501).

Für r, g und b muss jeweils ein ganzzahliger Wert zwischen 0 und 255 angegeben sein. Werte außerhalb dieses Intervalls werden als Wert 0 behandelt.

Farbnamen (colorname) und die RGB Standardwerte:

Black	0,0,0
Blue	0,0,255
Brown	144,48,0
Cyan	0,255,255
Darkblue	0,0,170
Darkcyan	0,146,170
Darkgreen	0,146,0
Default	0,0,0
Gray	131,131,131
Green	0,255,0
Magenta	255,0,255
Medium	255,255,255
Mustard	196,160,32
Orange	255,128,0
Purple	170,0,170
Red	255,0,0
White	255,255,255
Yellow	255,255,0

-cp:<pfad>

Pfad für Code Pages und die Datei mapping.def.

-csf:<n>

Faktor für die Glättung von Kurven. Je niedriger der Wert, desto glatter die Kurven. Möglich ist ein Wert von 1 bis 65535. Gilt nur für die Konvertierung nach AFP.

Beispiel: -csf:64

-dc:<n>

Standardwert: 64

-dc:<n>

Erzeuge Unterverzeichnisse und schränke die Ausgabe auf maximal <n> Dokumente pro Unterverzeichnis ein.

Diese Funktion gilt nur zusammen mit der Einstellung:

-DOC_INDEX oder mit

-DOC_COLD:XML, bzw. DOC_COLD:CSV.

Beispiel:

Die Option -dc:500 zusammen mit -doc_cold:xml schränkt die Anzahl von Dateien auf 1000 pro Unterverzeichnis ein: Sie erhalten in jedem Unterordner jeweils 500 Ausgabedokumente und 500 Indexdateien.

-dcp:<codepage>

Definiert die Default Code Page für die Konvertierung von Ressourcennamen, NOPs und Indizes in der AFP Spooldatei zu ASCII. (Beispiele: Index Namen und Werte, SFI Namen wie Doc Name, Page Name, Ressourcename, NOPs.)

Die mapping.def ordnet diese AFP Code Page zu der Code Page Datei (CP file). Der Standardwert wird somit in der mapping.def gesetzt.

-ed:<docid>

Das letzte zu konvertierende Dokument. Als <n> ist die Dokument ID anzugeben, wie diese aus dem Protokoll zu ermitteln ist.

Nur möglich, wenn DOC_INDEX oder DOC_COLD ausgewählt ist.

-ell:<n>

Detaillierungsgrad für Fehlermeldungen festlegen.

Eine der folgenden Optionen ist möglich:

1	Nur die Fehlermeldung (Standardwert)
2	Dateiname, Zeilennummer, Fehlermeldung

-fd:<formdef>

Standard FORMDEF, die für die AFP-Datei zu verwenden ist

-fdp:<pfad>

Pfad zu FORMDEF Ressourcen.

Es ist auch möglich, eine Liste von Pfaden anzugeben mit Komma als Trennzeichen.

Beispiel: samples/resource,c:\afp\formdef

-fnpt:<pattern>,Keyword1,Keyword2...

Entspricht dem INI Parameter FilenamePattern und definiert das Muster für Dateinamen mit Hilfe von Zeichenfolgen und/oder Platzhalter für reservierte Keywords.

Die Syntax:

-fnpt:

<pattern>,Keyword1,Keyword2...

<pattern> definiert das Formatierungsmuster und entspricht der Syntax eines C printf Befehls.

Soll ein Pattern einen TIMESTAMP Wert (siehe unten) formatieren, so verwenden Sie die Syntax eines C strftime Befehls. Die Keywords, die Sie angeben, bilden den Dateinamen entsprechend dem vorgegebenen Pattern. Der Datentyp eines Keywords muss mit dem Datentyp im Pattern übereinstimmen.

Mögliche Keywords:

DOCID	Dokument ID (Datentyp: Integer)
PAGEID	Seiten ID (Datentyp: Integer)
PID	Prozess ID (Datentyp: String)
TIMESTAMP[:<pattern>]	Aktueller Zeitstempel (Datentyp: String)
SPOOLNAME	Name der Spooldatei (Datentyp: String)

Standardwerte:

Das Standardformat für TIMESTAMP ist "%y%m%d%H%M%S"

Welcher Standardwert für -fnpt verwendet wird, hängt von den Einstellungen der INI Parameter FileCreationMode und GenerateUniqueFile (bzw. von den entsprechenden Kommandozeilenoptionen) ab wie folgt:

<i>Bedingung</i>	<i>Standardwert für -fnpt, bzw. FilenamePattern</i>
FileCreationMode ist entweder FILE_MERGE, DOC_INDEX, DOC_COLD, DOC, oder PAGE	"%s_%s.%d",SPOOLNAME,PID,DOCID
GenerateUniqueFile=on	"%s_%s.%d",SPOOLNAME,PID,DOCID
FileCreationMode=ALL und GenerateUniqueFile=off	"%s",SPOOLNAME
PageOutput=on (bzw. bei Verwendung der Kommandozeilenoption -po)	"%s_%s.%ld.%ld",SPOOLNAME,PID,DOCID,PAGEID

-fp:<pfad>

Pfad für Type 1 und TrueType Schriften.

-h or -?

Hilfe anzeigen. Das Programm zeigt die Liste der möglichen Kommandozeilenoptionen.

-iASCII

für die gegebene Datei jeweils einen optimalen Wert für die Kommandozeilenoptionen -mc und -mr ausgeben. Gilt nur für ASCII als Ausgabeformat.

-if:<infile>

Pfad und Dateiname der INI-Datei

-ip:<pfad>

Pfad zur Datei afp2web.ini

-jq:n

JPEG Qualität, mit n zwischen 1 und 100. 1 höchster Kompressionsgrad bei geringster Qualität, 100 ist der geringste Kompressionsgrad bei bester Qualität. Empfohlen wird als Wert 50.

-l

LOG-Datei (Teil 1: Parsing-Teil, Teil 2: Builder-Teil mit Schriftinformationen)

Hinweis: Wenn -ll:0 folgt, dann wird -l deaktiviert. Wenn Sie das vermeiden wollen, stellen Sie -l nach dem Parameter -ll oder vermeiden Sie -ll:0.

-lf

LOG-Datei (Teil 2: Builder-Teil mit Schriftinformationen)

Hinweis: Wenn -ll:0 folgt, dann wird -lf deaktiviert. Wenn Sie das vermeiden wollen, stellen Sie -lf nach dem Parameter -ll oder vermeiden Sie -ll:0.

-ll:<nn>

Schränkt die Meldungen in der LOG-Datei ein. Möglich für <nn> ist ein Wert aus:

0	Keine LOG-Datei (Standardwert)
1	Structured Fields (SFI)
2	Data Information der SFI
3	Repeating Groups
4	Patterns (wie Grafik)
5	FOCA Information
6	Mapping.def Zeilen am Ende der LOG-Datei einfügen. Diese Zeilen können als Vorlage für die Anpassung der mapping.def verwendet werden.
8	FTP Logging
ALL	Alle Information im FOCA
FONT	Builder Teil, Schriftinformation (entspricht LoggingFont=on)
Ressource Logging Levels	

RES	Alle Ressourcentypen
	oder eine der folgenden Ressourcentypen:
CDP	Code Pages
FDEF	Formdefs
FNTR	Fonts (Detaillierte Auflistung aller Schriften in der aktuellen Spooldatei)
IOB	Object Containers
MMAP	Medium Maps
OLY	Overlays
PSEG	Page Segments
Grafik Logging Levels	
INFO	Info-Meldungen aus der Grafikverarbeitung
WARN	Warn- und Info-Meldungen aus der Grafikverarbeitung
DEBUG	Warn-, Info- und Debugmeldungen aus der Grafikverarbeitung

Hinweis: Wenn -lf nach -ll:0 folgt, dann gilt -lf.

-lp:<pfad>

Pfad für die Ausgabe der LOG-Dateien.

-mc:<max columns>

-mr:<max rows>

Maximaler Wert für die Anzahl von Spalten und Zeilen auf einer Seite. Diese Parameter schränken die Größe einer Seite ein. Gilt nur für ASCII als Ausgabeformat.

-nocmyktorgb

Standardmäßig ist der INI Parameter CMYKTORGb aktiviert. Das bedeutet, für die Ausgabe nach PDF werden CMYK Farbenen zu RGB konvertiert. Mit dieser Kommandozeilenoption können Sie dies auch abschalten.

Hinweis: Bitte verwenden Sie diese Option nur für Druckdaten. CMYK für PDF führt zu große PDF Dateien. Wenn Sie Dokumente für die online Darstellung benötigen, verwenden Sie bitte die Konvertierung zu RGB.

-op:<pfad>

Pfad für die Ausgabedateien.

-os:<breite>,<höhe>

Größe der resultierenden Ausgabedatei. Breite und Höhe der Seite werden in Pixeln angegeben. Ein Anwendungsbeispiel ist die Erzeugung von Miniaturansichten.

Standardwert: Das Dokument wird in der Originalgröße ausgegeben.

-ovp:<pfad>

Pfad für Overlay-Ressourcen.

Es ist auch möglich, eine kommagetrennte Liste von Pfaden anzugeben.

Beispiel: samples/resource,c:\afp\overlays

-p

Protokoll ausgeben. Option ist nur möglich mit DOC_INDEX (oder DOC_COLD).

-po

Eine Datei pro Seite des Ausgabedokuments erstellen. Gilt nur für TIFF als Ausgabeformat.

-pp:[fp][-tp]

Seitenauswahl von(-bis)

-pr:<resolution>

Gilt nur für TIFF. Auflösung in dpi. Geben Sie einen der folgenden Werte ein:

-pr:200

-pr:240

-pr:300

-plw:<n>

Minimale Linienstärke, um zu gewährleisten, dass Linien ausgedruckt werden. Anzugeben in Pixeln. Gilt für die Konvertierung nach AFP.

Beispiel: PrintableLineWidth=1

Standardwert: 1

-ps:[<Owner Password>],[<User Password>][,Key Length[,Permission Flags]]

PDF Sicherheitseinstellungen

Die Parameter sind fest vorgegeben und haben die folgende Bedeutung:

Owner Password	Optional Owner Password. Passwort für die Änderungen der Sicherheitseinstellungen
User Password	Optional, User password, erforderlich für das Öffnen des PDF Dokuments.
Key Length	Optional. Verschlüsselungsstärke. Möglich ist ein Vielfaches von 8 zwischen 40 und 128 40 => Normale Verschlüsselung, 128 => Bessere Verschlüsselung Standard ist 40.
Permission Flags	Optional. Eine oder mehrere Schalter für Berechtigungen, jeweils getrennt mit " ". Als Schalter ist ein Wert aus der folgenden Liste zu nehmen. Wenn Schlüssellänge 40, so sind die Schalter 5,6,7,8 nicht erlaubt.

Die folgende Werte sind als Permission Flags möglich:

0	Keine Berechtigungen (Standardwert)
1	Drucken in niedriger Auflösung zulässig
2	Dokumentänderungen zulässig
3	Dokumentinhalt kopieren oder extrahieren zulässig
4	Kommentare eingeben und das Ausfüllen von Formularfeldern zulässig
5	Eingabe in Formularfelder oder von Signaturen zulässig
6	Inhalt für Ausgabehilfe entnehmen zulässig
7	Dokumentzusammenstellung zulässig
8	Drucken in hoher Auflösung zulässig

-psp:<pfad>

Pfad zu Page Segment Ressourcen.

Es ist auch möglich, eine kommagetrennte Liste von Pfaden anzugeben.

Beispiel: samples/resource,c:\afp\pagesegments

-

pui:[<HideMenubar>][,<HideToolbar>][,<HideWindowUI>]

PDF Betrachtereinstellungen ("User Interface")

Für jeden Positionsparameter, geben Sie entweder "on" oder "off" (ohne die Anführungszeichen) ein.

Der Standardwert ist "off" für alle Parameter. .

HideMenubar	on = Menüleiste wegblenden
HideToolbar	on = Werkzeugleiste wegblenden
HideWindowUI	on = Fensterkontrollen wegblenden

Hinweis: Wird ein Wert weggelassen, so gilt der Standardwert. Beispiel: -pui:,on,on entspricht -pui:off,on,on

Beispiele:

Nur die Menüleiste wegblenden: -pui:on

Menüleiste und Fensterkontrollen wegblenden: -pui:on,,on

-pv

Vorschau. Startet das geeignete Betrachterprogramm für die resultierende Ausgabedatei (den Adobe Reader für PDF oder einen Betrachter für eine ASCII, TIFF oder JPEG Datei).

-

pwn:[FitWindow][,<CenterWindow>][,<FullScreenWindow>] [,<DisplayDocTitle>]

PDF Betrachtereinstellungen für die Darstellung im Fenster.

Für jeden Positionsparameter, geben Sie entweder "on" oder "off" (ohne die Anführungszeichen) ein.

Der Standardwert ist "off" für jeden Parameter:.

FitWindow	on = Dokumentgröße in das Fenster einpassen
CenterWindow	on = Dokument im Fenster zentrieren
FullScreenWindow	on = Dokument im Vollbildschirmmodus zeigen, ohne Menüleiste, Fensterkontrollen oder sonstige Fenster.
DisplayDocTitle	on = Den Wert der Dokumenteigenschaft Title in der Fenstertitelleiste zeigen off = Die PDF Dateiname in der Fenstertitelleiste zeigen

Hinweis: Ein leerer Wert ist erlaubt und in diesem Falle gilt der Standardwert. Beispiel: -pwn:.,on,on,on entspricht -pwn:off,on,on,on.

-q

Meldungen an die Systemkonsole unterdrücken.

-r:<rotation>

Seitendrehung:

0	0 Grad im Uhrzeigersinn
90	90 Grad im Uhrzeigersinn
180	180 Grad im Uhrzeigersinn
270	270 Grad im Uhrzeigersinn
portrait	Hochformat
landscape	Querformat

-rf:<resourcefile>

Pfad und Dateinamen zu AFP Ressourcen..

-rp:<pfad>

Pfad zu AFP Ressourcen.

-S

Ausgabe einer Verarbeitungs-Statistik (stat*.txt)

-sa

Für die Verwendung in der AFP2web Scripting Facility. Argumente, die an das Skript zu übergeben sind.

-sd:<docid>

Das erste zu konvertierende Dokument. Als <n> ist die Dokument ID anzugeben, wie diese aus dem Protokoll zu ermitteln ist.

Nur möglich, wenn DOC_INDEX oder DOC_COLD ausgewählt ist.

-si

Lesen von Standardeingabe (nur für AFP).

-so

Schreiben nach Standardausgabe (nur für AFP).

-sod

Objektdaten auf Festplatte schreiben.

Legt fest dass Objektdaten temporär für die Verarbeitung im Pfad laut "TempPath" abgelegt werden. Diese Option ermöglicht es, Arbeitsspeicher einzusparen und die Ausgabegröße der Objekte zu optimieren.

-sp

Für die Verwendung in der AFP2web Scripting Facility.

Dateiname mit der Skriptprozedur.

-std

Lesen von Standardeingabe. Schreiben nach Standardausgabe (entspricht beide Optionen -si und -so).

-Strict

Konvertierung abbrechen, wenn eine erforderliche AFP Ressource nicht gefunden werden kann.

-tp:<pfad>

Pfad für temporäre Dateien. Der Pfad muss vorhanden sein.

AFP2web muss die Lese-/Schreibberechtigung für dieses Verzeichnis haben. Wenn nicht, so bricht AFP2web ab beim Versuch, temporäre Dateien zu schreiben oder zu löschen.

-u

Eindeutige Dateinamen für Ausgabedateien erzeugen. Dieser Parameter wird wegen der Rückwärtskompatibilität zu früheren Versionen weiter angeboten. Dieser Parameter ist nicht erforderlich für AFP2web ab der Version 3.x und wir raten von der Verwendung dieses Parameters ab.

-V

Programmversion von AFP2web anzeigen.

-vall

Versionen aller AFP2web Komponenten anzeigen.

**-wm:<text,
typeface,height,rotation,redmask,greenmask,
bluemask>**

Stellt Text als Wasserzeichen hinter dem Dokumenttext dar. Die möglichen Parameter:

text	Text
typeface	Schriftart
height	Schriftgröße in Zehntel Punkt. Geben Sie 90 für 9 Punkt.
rotation	Rotationswinkel. Ein Wert von 0 bis 360.
redmask	RGB Red Wert von 0 bis 255
greenmask	RGB Green Wert von 0 bis 255
bluemask	RGB Blue Wert von 0 bis 255

Beispiel:

-wm: "Confidential , Helvetica, 600, 60, 189, 219, 231"

Standardwert: kein Wasserzeichen

-xf:<afpindex>

Eingabedatei mit dem AFP Index zum AFP Dokument.

-xp:<pfad>

Pfad für die Ausgabe von Indexdateien.

5.4 Parameter der mapping.def

Funktion und Struktur

Die Datei mapping.def ist eine Textdatei im Unterverzeichnis /afpcp/ und dient generell als Konfigurationsdatei für die Verarbeitung oder Ersetzung von Schrift.

Bevor wir die einzelnen Abschnitte im Detail beschreiben, hier ein Überblick über die Abschnitte und deren Funktion:

Definiert den Standard Modus für die Verarbeitung von Schrift:

[CHARSET RENDERING]	Modus für die Schriftverarbeitung oder -ersetzung.
----------------------------	--

Abschnitte, die Standardattribute für AFP Schriften definieren. Angaben in diesen Abschnitten sind nur dann erforderlich, wenn AFP2web die Informationen nicht direkt aus dem AFP Input ermitteln kann:

[XFONT]	AFP Character Set, AFP Code Page
[CHARSET]	AFP Character Set, Font Global ID
[FGID]	Font Global ID, AFP Schriftnamen

Zu jedem AFP-Schriftnamen geben Sie in den folgenden Abschnitten an, welche Schrift für die Konvertierung zu verwenden ist. Sie können mehrere Schriften aufführen. Diese werden als Alternativen verwendet, wenn eine Schrift nicht im System zu finden ist:

[PDFFONT]	Schriftnamen für die Konvertierung nach PDF.
[WINFONT]	Schriftnamen für die Konvertierung zu einem Rasterformat (TIFF/JPEG/...). Die Schrift muss im Windows Betriebssystem (Ordner für Schriftarten) installiert sein.
[UNIXFONT]	Schriftnamen für die Konvertierung zu einem Rasterformat (TIFF/JPEG/...). Die Schrift muss im Unix Betriebssystem installiert sein
[FONTSUFFIX]	Namenskonvention für Schriftdateien.

Definiert die zu verwendenden AFP Schriften:

[AFPFONT]	Zuordnung einer PDF Schrift zu AFP Character Set und Code-page
------------------	--

Der folgende Abschnitt definiert die Zuordnung von Codes Pages und Character Sets:

[CODEPG]	Zuordnung AFP Code Pages zu Code Page Dateien
-----------------	---

Hinweise für die Verwendung

Sie passen die Datei mapping.def für Ihre Zwecke an. Sie lassen die Standardeinträge stehen und ergänzen diese um Ihren zusätzlichen Anforderungen. Wenn Sie mehrerer Regeln definieren, so beachten Sie bitte: Die erste treffende Regel oben in der Liste ersetzt alle nachfolgenden Regeln.

Wenn Sie Einträge in der mapping.def verändern, beachten Sie Folgendes:

- Eine Zeile mit einem Semikolon (;) als erstes Zeichen gilt als Kommentar.
- Jeder Abschnitt ist unterteilt in Einträge für „;user defined“ und solche unter „;standard“. Eine Definition, die Sie unter „;user defined“ hinzufügen, ersetzt eine gleich lautende Definition unter „;standard“.

Wichtig: Bitte ändern Sie unter keinen Umständen die Einträge unter „standard“!

- Wenn Sie AFP Fontressourcen für AFP2web zur Verfügung stellen, sind Einträge in den Abschnitten [XFONT], [CHARSET], [FGID] nicht mehr erforderlich.
- Die mapping.def erlaubt eine maximale Zeilenlänge von 1 K.

[CHARSET RENDERING] Abschnitt

Syntax, Beispiel

Syntax:

charsetname=RenderingFlag, option

Beispiel: COH000A0=2

Funktion

Bestimmt für einzelne AFP Character Sets den Standardmodus für die Verwendung oder Einbettung von Schrift.

Parameter

Parameter	Beschreibung
charsetname	AFP Character Set. Wenn erforderlich, so ist das AFP Character Set im Abschnitt [CHARSET] zu definieren. <i>Hinweis: Wild cards sind als Platzhalter möglich. (Siehe die Verwendungshinweise unten.).</i>
RenderingFlag	Einer der folgenden Werte: 0 Font Rasterizing (Standard) Schriftrasterung. Dieser Modus verwendet die originalen Schriften. Dieser Modus ist der Standardmodus für AFP2web. 1 Font Referencing/Substituting (Ersatzschrift verwenden). <ul style="list-style-type: none">• PDF: Referenziere die 14 Standardschriften für PDF oder die unterstützte externe Ersatzschrift• TIFF: Verwende die Ersatzschrift 2 Font Embedding (Ersatzschrift verwenden) <ul style="list-style-type: none">• PDF: Ersatzschrift in PDF einbetten• TIFF: Nicht relevant

Parameter	Beschreibung
option	<p>Weitere Optionen für die Konvertierung von PDF zu AFP. Gilt nur für RenderingFlag=0.</p> <p>Hinweis: Eine als "geplant" gekennzeichnete Option ist für eine spätere Version von AFP2web vorgesehen.</p> <p>Ein Wert aus:</p> <p>0 Schrift unverändert übernehmen (Standardeinstellung). AFP2web verwendet die Schrift aus der PDF Eingabe.</p> <p>Type 3 Rasterschriften werden zu AFP Rasterschriften</p> <p>geplant: Type 1 und TrueType Schriften werden zu AFP Outline Fonts</p> <p>1 Inline Image: AFP2web konvertiert, bzw. rastert PDF Type 1 und Truetype Schriften im Text zu IOCA Images.</p> <p>2 geplant: AFP Raster Font: AFP2web wandelt Type 1 und TrueType Schriften zu AFP Rasterschriften.</p> <p>3 geplant: AFP Outline Font: AFP2web wandelt Type 1 und TrueType Schriften zu AFP Outline Fonts.</p>

Verwendung

Für die Ausgabe nach PDF bei RenderingFlag 0 (Schrift rasterung) wird die verwendete Teilmenge der Schrift in PDF eingebettet.

Für die Darstellung der PDF Datei empfehlen wir die Verwendung des Adobe Acrobat Readers ab der Version 7.

Die Verwendung von Wildcards für Character Sets

Als Wild Cards (Platzhalterzeichen) für die Namen von Character Sets möglich:

Wild Card Zeichen	Bedeutung
*	1 oder mehrere Zeichen
?	Genau 1 Zeichen an dieser Stelle

Hinweis: Bitte verwenden Sie Platzhalterzeichen mit Vorsicht. AFP2web liest die Einträge von oben nach unten - der erste Treffer wird genommen.

Beispiel:

```
[CHARSET RENDERING]
; Syntax: <CharsetName>=<RenderingFlag>
; RenderingFlag=>Values
; 0=>Rasterize
; 1=>Reference / Substitute
; 2=>Embed
;
C?H400B0=1
C?H400*=2
```

Ergebnis:

- Fall 1: Das Character Set "C0H400B0" passt zu beiden Mustern "C?H400B0=1", d.h. "C<beliebiges Zeichen>H400B0" und "C?H400*=2", d.h. "C<beliebiges Zeichen>H400<beliebige Folge von Zeichen>". AFP2web verwendet den ersten passenden Eintrag mit der RenderingFlag 1.
- Fall 2: Das Character Set "C4H40090" passt zum zweiten Muster "C?H400*=2", d.h. "C<beliebiges Zeichen>H400<beliebige Folge von Zeichen>". AFP2web verwendet den zweiten Eintrag mit der RenderingFlag 2.

[XFONT] Abschnitt

Syntax, Beispiel

Syntax:

`xfont = characterset, codepage`

Beispiel: `X0GB02=C0D0GB10,T1V10273`

Funktion

Legt AFP Character Set und AFP Code Page eines Coded Fonts fest

Parameter

<i>Parameter</i>	<i>Beschreibung</i>
xfont	Coded Font (Siehe die Anmerkungen unten.)
characterset	AFP Character Set (Siehe dazu die Anmerkungen unten.) Der AFP Character Set muss im [CHARSET] Abschnitt vorhanden sein.
codepage	AFP Code Page Der AFP Code Page muss im [CODEPG] Abschnitt vorhanden sein.

Anmerkungen zu xfont, characterset

Das zweite Zeichen im Namen eines Coded Fonts oder Character Set weist auf eine Rasterschrift (0 oder 1) oder auf eine Vektorschrift (Z) hin. Beispiel: Eine Rasterschrift liegt vor, wenn der Name des Coded Fonts mit XO... oder X1... beginnt, bzw. den eines Character Sets mit C0... oder C1....).

Die Regel ist:

- Ist das zweite Zeichen ?, so sucht AFP2web nach einer Rasterschrift und nicht nach einer Vektorschrift.
- Ist das zweite Zeichen 0 oder 1 (Rasterschrift), so sucht AFP2web nach diesem Wert. Wird die Schrift nicht gefunden sucht AFP2web alternativ nach 0 oder 1.
- Das zweite Zeichen muss Z sein, wenn AFP2web nach einer Vektorschrift suchen soll.

Verwendung

Sie legen diese Angaben fest, wenn Sie ihre eigenen Coded Fonts verwenden oder wenn Sie Coded Fonts ändern. Fügen Sie Ihre Einträge stets unter „user-defined“.

Bitte ändern Sie unter keinen Umständen die Einträge unter „standard“.

Das Resultat ist eine eingebettete Teilmenge der Schrift in PDF.

Für die Darstellung der PDF Datei empfehlen wir die Verwendung des Adobe Acrobat Readers ab der Version 7.

[CHARSET] Abschnitt

Syntax, Beispiel

Syntax:

```
charset = fgid, height, width, strikeover, underline
```

Beispiel: CODOGB10=39,120,144,0,0

Funktion

Zuordnung eines Namen des AFP Character Sets zu einer Font Global ID.

Legt weitere Schriftattribute fest (wie Schrifthöhe, Laufweite).

Parameter

Parameter	Beschreibung
charset	AFP Character Set
fgid	Font Global ID Muss im Abschnitt [FGID] definiert sein.
height	Mittlere Schrifthöhe in zehntel Punkt Ein Wert von 1 bis 999 Beispiel: Geben Sie 90 für eine Schrifthöhe von 9 Punkt.

<i>Parameter</i>	<i>Beschreibung</i>
width	Mittlere Laufweite in zehntel Punkt. Ein Wert von 1 bis 999 Beispiel: Geben Sie 90 für eine mittlere Laufweite von 9 Punkt. (Dieser Parameter ist für eine spätere Verwendung vorgesehen.)
strikeover	Text mit Durchstreichung 0 nein 1 durchgestrichen
underline	Unterstreichung 0 nein 1 unterstrichen

Verwendung

Sie legen diese Angaben fest, wenn Sie Ihre eigenen AFP Character Sets verwenden oder wenn Sie vorhandene AFP Character Sets verändern.

Fügen Sie Ihre Einträge stets unter „user-defined“. Bitte ändern Sie unter keinen Umständen die Einträge unter „standard“.

[FGID] Abschnitt

Syntax, Beispiel

Syntax:

`fgid = logicalname, style, weight, italic
Beispiel: 39 = GOTHIC,MODERN,BOLD,0`

Funktion

Zuordnung Font Global ID zu einem AFP Schriftnamen. Legt weitere Schriftattribute fest (Stärke, Schnitt).

Parameter

<i>Parameter</i>	<i>Beschreibung</i>
fgid	Font Global ID
logicalname	AFP Schriftnamen. Muss im Abschnitt [PDFFONT], [WIN-FONT], bzw. [UNIXFONT] definiert sein.
style	Schriftstil: SWISS Proportional, ohne Serifen ROMAN Proportional, mit Serifen SCRIPT monospace, dekorative 'Handschrift' MODERN monospace, mit oder ohne Serifen DISPLAY dekorativ Gilt nur für die Ausgabe von TIFF unter Windows
weight	Schriftstärke: BOLD fett MED normal LIGHT mager Das derzeitige Release von AFP2web kennt nur BOLD oder MED.
italic	0 normal 1 kursiv

Verwendung

Fügen Sie Ihre Einträge stets unter „user-defined“. Bitte ändern Sie unter keinen Umständen die Einträge unter „standard“.

[FONTSUFFIX] Abschnitt

Syntax, Beispiel

Syntax:

```
<Style>=<suffix1>[ , <suffix2> . . . , <suffixN>]
```

Beispiel: BoldItalicSuffix=bi,b,-BoldItalic,-BoldOblique

Funktion

Spezifiziert Dateinamen-Suffixe für Schriften.

Es gibt keinen allgemeingültigen Standard für die Dateinamen von Schriften. AFP2web geht standardmäßig von einer Konvention aus, bei der Suffixe für die Schriftattribute bold und italic verwendet werden. In diesem Abschnitt der mapping.def geben Sie an, welche Suffixe nach dieser Konvention in Ihrem System vorkommen.

Hinweis: Für Unix Systeme ist die Groß-/Kleinschreibung auch für Suffixe zu beachten.

Parameter

Parameter	Beschreibung
Style	Einer der folgenden Werte: BoldSuffix ItalicSuffix BoldItalicSuffix

Verwendung

Beispiel: Hat die Schrift Verdana die Attribute:BoldItalic, so geben wir an: BoldItalic-Suffix=b,-BoldItalic. AFP2web sucht dann in den angegebenen Verzeichnissen (Standard: pdffont\) zuerst nach Dateien mit dem Namen Verdana*. * und dann erst nach Dateien mit dem Namen Verdana-BoldItalic. * . Wird mehr als eine Datei gefunden (z.B. Verdana-BoldItalic.pfm, Verdana-BoldItalic.ttf), so nimmt AFP2web die erste Datei.

[PDFFONT] Abschnitt

Syntax, Beispiel

Syntax:

Logical name = Font, Alias_1, Alias_2, . . . , Alias_n

Beispiel: GOTHIC=Gothic,GothicText,CourierNew

Funktion

Dieser Abschnitt wird nur für die Konvertierung nach PDF verwendet. Die Konvertierung nach TIFF wird durch den Abschnitt [WINFONT] oder [UNIXFONT] festgelegt.

Legt für jeden AFP Schriftnamen die Schrift fest. Nennt weitere Schriften, die als Alternativen verwendet werden können.

AFP2web sucht von links nach rechts und verwendet die Schrift, die zuerst gefunden wird.

Parameter

<i>Parameter</i>	<i>Beschreibung</i>
logicalname	AFP Schriftname Referenziert den Wert im Abschnitt [FGID]
Font	Name der Ersatzschrift
Alias_1, Alias_2,..., Alias_n	Weitere alternative Schriften

Verwendung

Fügen Sie Ihre Einträge stets unter „user-defined“. Bitte ändern Sie unter keinen Umständen die Einträge unter „standard“.

[WINFONT] Abschnitt

Syntax, Beispiel

Syntax:

`logicalname = Font, Alias_1, Alias_2, . . . , Alias_n`

Beispiel: GOTHIC=Gothic,Gothic Text,Courier New

Funktion

Dieser Abschnitt wird nur für die Konvertierung nach TIFF verwendet. Die Konvertierung nach PDF wird durch den Abschnitt [PDFFONT] festgelegt.

Legt für jeden AFP Schriftnamen die in UNIX installierte Schrift fest.

Nennt weitere Windows Schriften, die als Alternativen verwendet werden können.

AFP2web verwendet die erste Windows-Schrift, die gefunden werden kann.

Parameter

Parameter	Beschreibung
logicalname	AFP Schriftnamen Referenziert den Wert im Abschnitt [FGID]
Font	Name der Windows Schriftfamilie Den Namen finden Sie im Ordner „Schriften“ der Windows Systemsteuerung.
Alias_1, Alias_2,..., Alias_n	Weitere alternative Schriften

Verwendung

Fügen Sie Ihre Einträge stets unter „user-defined“. Bitte ändern Sie unter keinen Umständen die Einträge unter „standard“.

[UNIXFONT] Abschnitt

Syntax, Beispiel

Syntax:

```
Logicalname = Font, Alias_1, Alias_2, . . . , Alias_n
```

Beispiel: GOTHIC=gothic,courier

Funktion

Dieser Abschnitt wird nur für die Konvertierung nach TIFF verwendet. Die Konvertierung nach PDF wird durch den Abschnitt [PDFFONT] festgelegt.

Legt für jeden AFP Schriftnamen die in UNIX installierte Schrift fest.

Nennt weitere Schriften, die als Alternativen verwendet werden können. AFP2web verwendet die erste Schrift, die gefunden werden kann.

Parameter

Parameter	Beschreibung
logicalname	AFP Schriftnamen Referenziert den Wert im Abschnitt [FGID]
Font	Name der Schriftfamilie
Alias_1, Alias_2,..., Alias_n	Weitere alternative Schriften

Verwendung

Fügen Sie Ihre Einträge stets unter „user-defined“. Bitte ändern Sie unter keinen Umständen die Einträge unter „standard“.

[AFPFONT] Abschnitt

Syntax, Beispiel

Syntax:

```
typeface[-stärke][neigung]=
AFPOutlineCodedFont|AFPOutlineCharacterSet, AFPCodePage
```

Beispiel:

```
CourierNew=CZ4200, T1001141
CourierNew-Italic=CZ4300, T1001141
CourierNew-Oblique=CZ4300, T1001141
CourierNew-Bold=CZ4400, T1001141
CourierNew-BoldItalic=CZ4500, T1001141
```

Funktion

Dieser Abschnitt wird nur für die Konvertierung nach AFP verwendet.

Legt für jeden Typefacenames die zu verwendende AFP Schrift fest. .

Parameter

<i>Parameter</i>	<i>Beschreibung</i>
typeface	Name der Schriftfamilie (Typeface Name)
stärke	Schriftstärke: Bold oder Light
neigung	Schriftschnitt: Oblique oder Italic
AFPOutlineCodedFont	Name des AFP Outline Coded Fonts
AFPOutlineCharacterSet	Name des AFP Outline Character Sets
AFPCodePage	Name des AFP Code Page

Verwendung

Fügen Sie Ihre Einträge stets unter „user-defined“. Bitte ändern Sie unter keinen Umständen die Einträge unter „standard“.

[CODEPG] Abschnitt

Syntax, Beispiel

Syntax:

```
codepage = cpgid, wincp  
Beispiel: T1V10273=273,ANSI
```

Funktion

Zuordnung einer AFP-spezifischen Code Page ID zum Windows-spezifischen ANSI Code Page.

Parameter

<i>Parameter</i>	<i>Beschreibung</i>
codepage	AFP Code Page

<i>Parameter</i>	<i>Beschreibung</i>
cpgid	Name der Code Page Datei (bitte ohne die Dateierweiterung .cp) Für ASCII-encodierte AFP Daten geben Sie 819 für die Standard Map Datei 819.cp an.
wincp	Einer der Windows Zeichensätze: ANSI Standard Character Set SYMBOL NULL Built-in encoding

Verwendung

Wenn Sie ihre eigenen AFP Code Pages erstellen oder vorhandene AFP Code Pages verändern, so müssen Sie:

- Die zugehörige Code Page Datei (<codepage ID>.cp) erstellen und in das hierfür vorgesehene Verzeichnis einstellen (afpcp/, bzw. das Verzeichnis wird in der afp2web.ini festgelegt oder mit der entsprechenden Kommandozeilenoption). Bitte konsultieren Sie Maas High Tech Software GmbH für weitere Information zu diesem Verfahren.
- Eine neue Zeile unter [CODEPG] in der mapping.def eintragen

Fügen Sie Ihre Einträge stets unter „user-defined“. Bitte ändern Sie unter keinen Umständen die Einträge unter „standard“.

5.5 Code Page Dateien (CP-Dateien)

Eine Code Page Datei (CP-Datei) definiert für jeden Graphic Character Global Identifier (GCGID) den zu verwendenden EBCDIC, ASCII und Unicode Zeichencode. AFP2web verwendet diese Angaben wie folgt:

- Unicode wird als Zeichencode bei der Konvertierung von Schrift von AFP zu PDF verwendet. Die Kodierung in Unicode ermöglicht die Volltextsuche in den PDF Dateien.
- AFP2web verwendet ASCII Codes für die Namen von AFP Ressourcen und für AFP Indexelemente. Die AFP2web Scripting Facility verwendet grundsätzlich ASCII.
- Für die Verwendung einer Ersatzschrift ist zu gewährleisten, dass die Kodierung einer Schrift erhalten bleibt. Beispiel: AFP Schriften haben EBCDIC Zeichencodes, PC-Schriften hingegen ASCII. Die CP-Dateien regeln die Zuordnung von EBCDIC Codes zu ASCII Codes.
- Möglicherweise verwenden Sie spezifische Zeichentabellen oder Ihre AFP Fontressourcen verwenden nicht die Standard Graphic Character Global Identifiers (GCGIDs) zur Identifizierung einzelner Zeichen im Character Set. In diesem Falle definieren Sie in den CP-Dateien logischen Namen und die Codes für die Zeichen in einem AFP Character Set.

Sie erstellen eine neue angepasste CP-Datei nur, wenn Sie spezifische, angepasste Code Pages oder wenn Ihre AFP Fontressourcen nicht die Standard IBM Namen für die Identifizierung von Zeichen in dem AFP Character Set verwenden (Graphic Character Global Identifiers GCGID).

Es ist auch möglich, für die Darstellung eines Zeichens eine bestimmte Glyphe aus der Schrift zu verwenden. Hierzu wird der Name der Glyphe als weiteren Parameter in der Code Page Datei eingetragen. Der Glyphen-Name ist in der Schriftdatei vorgegeben.

Im Zweifelsfall beantwortet Ihr Schriftlieferant die Frage nach der verwendeten Zeichenkodierung.

Das Format der Code Page Datei (CP-Datei mit der Dateierweiterung *.cp):

```
<GCGID> <EBCDICValue> <UnicodeValue> <ASCIIValue>[<GlyphName>]
```

Erläuterungen:

- GCGID Namen zur eindeutigen Identifizierung eines Zeichens.
- Der EBCDIC und ASCII Werte sind jeweils zweistellig, hexadezimal.
- Der Unicode Wert ist 4-stellig, hexadezimal.
- (Optional) Name des Glyphs wie dieser in der Schriftdatei vorkommt

Beispiel:

```
LA020000 C1 0041 41
LB020000 C2 0042 42
LC020000 C3 0043 43
; ohungarumlaut
```

L0250000 CF 0151 F5 ohungaruml aut

Wenn die Unicode Werte in einer Code Page Datei (CP-Datei) fehlen, verwendet AFP2web die Standardvorgabe aus der Datei gcgid2unicode.def. Sie finden diese Datei im Verzeichnis für Code Pages (/afpcp/) der AFP2web Installation. gcgid2unicode.def definiert die Zuordnung von IBM Standard GCGIDs zu Unicode. Informationen zu GCGIDs finden Sie in der "Technical Reference for IBM Expanded Core Fonts (S544-5228-01)" der IBM.

5.6 Die Ausgabe von Schriftinformationen in der LOG-Datei

Sie verwenden den INI-Parameter **LoggingLevel=FONT**, **LoggingFont=on** oder die Kommandozeilenoption **-lf**, um eine LOG-Datei zu erstellen mit einer Auflistung aller Schriften in der AFP Spooldatei.

Das Format der LOG-Meldungen zur Schrift

Eingabeschriften

Informationen über die Schrift der Eingabedatei beginnt mit der Zeichenfolge "==" wie folgt:

```
==>[CF=<Coded Font Name>,[ CS=<Char Set Name>,[ FGID=<FGID>,[
TF=<Type Face Name>,[ W=M|L|B|I], S=<Size>,[ SF|OT|TT|T1|RF|OF|,
CP=<CodePageName> | CPGID=<Code Page Global ID>],[ ENC=<Encoding>]
```

Ausgabeschriften

Informationen über die Schrift der Ausgabe beginnt mit der Zeichenfolge "-->", wie folgt:

```
-->[CF=<Coded Font Name>,[ CS=<Char Set Name>,[ FGID=<FGID>,[
TF=<Type Face Name>,[ W=M|L|B|I], S=<Size>,[ SF|OT|TT|T1|RF|OF|RI|,
CP=<CodePageName> | CPGID=<Code Page Global ID>],[ ACP=<CP
Source>: <CP File>],[ ENC=<Encoding>]
```

Text, für den die Schriftattribute gelten

Der Text, für den die Schriftattribute gelten, wird innerhalb eckiger Klammern ausgegeben:

```
>some text      <
>some text      <
```

Verwendete Kürzel in den LOG-Meldungen zur Schrift

Die folgende Tabelle beschreibt die Kürzel, die in den LOG-Meldungen zur Schrift verwendet werden:

<i>Kürzel</i>	<i>Beschreibung</i>
ACP	ASCII Code Page. "ACP" weist darauf hin, dass eine CP-Datei für die AFP Code Page verwendet wird, und gibt an, wie der Name der CP-Datei ermittelt wird. Gezeigt wird auch die verwendete Encodierung.
CF	AFP Coded Font Name. Gilt nur für AFP als Eingabe- oder Ausgabeformat. Für AFP als Eingabeformat, wird diese Information nur ausgegeben, wenn der Schrifttyp "Coded Font" verwendet wird.
CP	AFP Code Page. Nur für AFP. Für AFP als Eingabeformat: Nur wenn der Schrifttyp "Coded Font" oder "Character Set", jedoch nicht wenn "FGID" verwendet wird. Für AFP als Ausgabeformat: Nur, wenn der Schrifttyp "Coded Font" oder "Character Set" verwendet wird.
CP File	Name der ASCII Map Code Page (.cp) Datei

Kürzel	Beschreibung
CP Source	<p>CP Source gibt an, wie die CP-Datei ermittelt wird. Die Kürzel:</p> <p>MD CP Name der Code Page aus einem Eintrag in der mapping.def</p> <p>CP-DSC "Code Page Global ID", wie diese entweder in der "Code Page Descriptor" SFI oder in der "Map Coded Font" SFI angegeben ist.</p> <p>NC Ermittlung durch Namenskonvention (die letzten 4 Buchstaben des Code Page Namens, führende Nullen werden entfernt)</p> <p>MD-DEF Der Standardwert als "DEFAULT" in der mapping.def</p> <p>INI Die CP-Datei aus dem INI Datei Parameter "Code-Page"</p> <p>PG-DEF Der Standardwert aus dem Programm, AFP2web verwendet eine vordefinierte EBCDIC-TO-ASCII Array für die Umsetzung</p>
CPGID	AFP Code Page Global ID. Nur wenn die Eingabe-Schrift als "FGID" angegeben ist. Gilt nur für AFP als Eingabeformat.
CS	Character Set. AFP Character Set Name. Gilt nur für AFP. Für AFP als Eingabeformat: Nur wenn der Schrifttyp entweder als "Coded Font" oder "Character Set" angegeben ist, nicht jedoch als "FGID".
Emb	Schrift wird eingebettet. Gilt im Zusammenhang mit Angabe der Schriftformate OT, TT und T1

<i>Kürzel</i>	<i>Beschreibung</i>
ENC	<p>Die verwendete Enkodierung.</p> <p>For PDF als Eingabeformat: ANSI Windows ANSI Enkodierung ROMAN Macintosh ROMAN Enkodierung EXPERT Macintosh EXPERT Enkodierung STANDARD Adobe STANDARD Enkodierung SYMBOL Symbol Enkodierung NULL Die Schrift Built-in Enkodierung CUSTOM PDF spezifisch erzeugte Enkodierung Identity-H Standard Adobe CMAP Tabelle</p> <p>Für PDF und TIFF als Ausgabeformat: ANSI Windows ANSI Enkodierung SYMBOL Symbol Enkodierung NULL die Schrift Built-in Enkodierung</p> <p>Für AFP als Ausgabeformat: ASCII ASCII Enkodierung EBCDIC EBCDIC Enkodierung</p>
FGID	AFP Font Global ID. Ausgegeben wenn die Eingabe den Schrifttyp als "Coded Font", "Character Set" oder "FGID" angibt. Gilt nur für AFP als Eingabeformat.
OF	Schrifttyp ist AFP Outline Font
OT	Schrifttyp ist Open Type
Ref	Schrift wird referenziert. Diese Information wird zusammen mit Angabe des Font Typs OT, TT und T1 ausgegeben.
RF	Schrifttyp ist AFP Raster Font
RI	Text gerastert als Inline-Grafik
S	Schriftgröße (Höhe) in Punkt

<i>Kürzel</i>	<i>Beschreibung</i>
SF	Schrifttyp ist Standard oder System (für Eingabe). Für PDF Ausgabe: Standard Font Für TIFF Ausgabe: System Font
T1	Schrifttyp ist Type 1
TF	Schrift Typeface Name.
TT	Schrifttyp ist TrueType
UsedFont	Für PDF und TIFF Ausgabe: Name der verwendeten Schrift Für AFP Ausgabe: Coded Font (und oder) Character Set Name
W	Schriftstärke und Schnitt. Möglich sind die Werte M (Medium), L (Light), B (Fett), I (Italic, Kursiv). Beispiele: W=B bedeutet Schriftstärke ist fett W=MI Schriftstärke Normal und Schriftschnitt ist kursiv.

Hinweis: Fehlt eine AFP Fontressource, so steht ein Sternzeichen (*) hinter dem Ressourcennamen. Beispiel: CS=C1H400B0(*), TF=...

Beispiele für LOG-Meldungen zur Schrift

LOG für Rasterschriften

PDF Ausgabe:

```
==>CF=X0A00550, CS=COA07580, FGID=2308, TF=SONORAN SANS SERIF,
W=M, S=8.00, RF, CP=T1GI0361
-->UsedFont=COA07580, TF=F1, W=M, S=8.00, RF, UsedCP=MD: 2065.cp,
Enc=ANSI
```

```
>Vos paiements sont acceptés à la plupart des banques ou <
>N° de compte <
```

Hinweis: "F1" ist der Name der Schrift aus der mpdf Bibliothek. Die Schrift wird in der PDF Datei mit diesem Namen referenziert.

TIFF Ausgabe:

```
==>CF=X0A00550, CS=COA07580, FGID=2308, TF=SONORAN SANS SERIF,
W=BI, S=8.00, RF, CP=T1GI0361
-->UsedFont=COA07580, TF=SONORAN SANS SERIF, W=BI, S=8.00, RF,
UsedCP=MD: 2065.cp, Enc=ANSI
```

```
>Vos paiements sont acceptés à la plupart des banques ou <
>N° de compte <
```

AFP Ausgabe (Coded Font):

```
==>CF=X0A00550, CS=NULL, FGID=2308, TF=SONORAN SANS SERIF, W=M,
S=8.00, RF, CP=NULL
-->CF=X0A00550, CS=NULL, FGID=2308, TF= SONORAN SANS SERIF, W=M,
S=8.00, CF,
CP= NULL, ACP=1141.cp, Enc=EBCDIC
```

```
>Vos paiements sont acceptés à la plupart des banques ou <
>N° de compte <
```

AFP Ausgabe (Character Set):

```
==>CF=NULL, CS=COA07580, FGID=2308, TF=SONORAN SANS SERIF, W=M,
S=8.00, RF, CP=T1GI0361
-->CF=NULL, CS=COA07580, FGID=2308, TF= SONORAN SANS SERIF, W=M,
S=8.00, CS, CP=T1GI0361, ACP=1141.cp, Enc=EBCDIC
```

```
>Vos paiements sont acceptés à la plupart des banques ou <
>N° de compte <
```

LOG für Outline-Schriften

PDF/TIFF Ausgabe:

```
==>CF=X0A00050, CS=COA00050, FGID=2308, TF=SONORAN SANS SERIF,
W=B, S=8.00, OF, CP=T1VI0500
```

```
-->UsedFont=COA00050, TF=SONORAN SANS SERIF, W=B, S=8.00, OF,  
UsedCP=NC:500.cp, Enc=ANSI  
>Vos paiements sont acceptés à la plupart des banques ou <  
>N° de compte <
```

AFP Ausgabe (Coded Font):

```
=>CF=XOA00050, CS=NULL, FGID=2308, TF=SONORAN SANS SERIF, W=B,  
S=8.00, OF, CP=NULL  
-->CF=NULL, CS=COA07580, FGID=2308, TF=SONORAN SANS SERIF, W=M,  
S=8.00, CP=NULL, ACP=1141.cp, Enc=EBCDIC  
>Vos paiements sont acceptés à la plupart des banques ou <  
>N° de compte <
```

AFP Ausgabe (Character Set):

```
=>CF=XOA00550, CS=COA07580, FGID=2308, TF=SONORAN SANS SERIF,  
W=M, S=8.00, OF, CP=T1GI0361  
-->CF=NULL, CS=COA07580, FGID=2308, TF=SONORAN SANS SERIF, W=M,  
S=8.00, CP=T1GI0361, ACP=1141.cp, Enc=EBCDIC  
>Vos paiements sont acceptés à la plupart des banques ou <  
>N° de compte <
```

LOG für die Schriftreferenzierung

Fall 1: Referenzierung der 14 Standardschriften für PDF, oder die Verwendung von Systemschriften für die Ausgabe nach TIFF

PDF/TIFF Ausgabe:

```
=>CF=XOH400B0, CS=C1H400B0, FGID=2308, TF=HELVETICA LATIN1, W=B,  
S=8.00, RF, CP=T1VI0500  
-->UsedFont=Helvetica-Bold, TF=Helvetica-Bold, W=B, S=8.00, SF,  
UsedCP=NC:500.cp, Enc=ANSI  
>Vos paiements sont acceptés à la plupart des banques ou <  
>N° de compte <
```

Fall 2: Referenzierung externer Type1/TrueType Schriftdateien

PDF/TIFF Ausgabe:

```
=>CF=XOH410DC, CS=COH410DC, FGID=2308, TF=Optima, W=B, S=8.00,  
RF, CP=T1VI0500  
-->UsedFont=C:\winnt\Optima-Bold.pfm, TF=Optima-Bold, W=B,  
S=8.00, T1, Ref, UsedCP=NC:500.cp, Enc=ANSI  
>Vos paiements sont acceptés à la plupart des banques ou <  
>N° de compte <
```

Fall 3: Referenzierung externer AFP Schriften

AFP Ausgabe (Coded Font)

```
=>CF=XOH400B0, CS=C1H400B0, FGID=2308, TF=HELVETICA LATIN1, W=B,  
S=8.00, RF, CP=T1VI0500
```

```
-->UsedFont= X0H400B0, TF= HELVETICA LATIN1, W=B, S=8.00, CF, Ref,
Enc=EBCDIC
>Vos paiements sont acceptés à la plupart des banques ou <
>N° de compte <
```

AFP Ausgabe (Character Set)

```
==>CF=X0H400B0, CS=C1H400B0, FGID=2308, TF=HELVETICA LATIN1, W=B,
S=8.00, RF, CP=T1VI0500
-->UsedFont= C1H400B0, TF= HELVETICA LATIN1, W=B, S=8.00, CS, Ref,
CP= T1VI0500, Enc=EBCDIC
>Vos paiements sont acceptés à la plupart des banques ou <
>N° de compte <
```

LOG für die Schrifteinbettung (Embedding)

PDF Ausgabe:

```
==>CF=X0H410DC, CS= C0H410DC, FGID=2308, TF=Optima, W=B, S=8.00,
RF, CP=T1VI0500
-->UsedFont=C:\winnt\Optima-Bold.pfb, TF=Optima-Bold, W=B,
S=8.00, T1, Emb, UsedCP=NC:500.cp, Enc=ANSI
>Vos paiements sont acceptés à la plupart des banques ou <
>N° de compte <
```

Für die Ausgabe nach TIFF ist die Schrifteinbettung nicht möglich. Die Ausgabe nach TIFF wird daher stets als "Font Referencing" behandelt.

Irrelevant für die AFP Ausgabe.

LOG für eine FGID Schrift

PDF/TIFF Ausgabe:

```
==>FGID=2307, TF=Helvetica, W=BI, S=8.00, CPGID=500
-->UsedFont=Helvetica-Bold-Italic, TF=Helvetica-Bold-Italic,
W=BI, S=8.00, SF, UsedCP=CPGID:500.cp, Enc=ANSI
>Vos paiements sont acceptés à la plupart des banques ou <
>N° de compte <
```

Irrelevant für die AFP Ausgabe

LOG für Text, der durch die Scripting Facility eingefügt wird

PDF/TIFF Ausgabe:

```
==>TF=Helvetica, W=BI, S=8.00
```

```
-->UsedFont=Helvetica-Bold-Italic, TF=Helvetica-Bold-Italic,  
W=BI, S=8.00, SF, Enc=ANSI  
>Vos paiements sont acceptés à la plupart des banques ou <  
>N° de compte <
```


5.7 Schriftzuordnung laut IBM Namenskonvention für Ressourcen

Überblick

Wenn Sie eine AFP Fontressource (Character Set oder Coded Font) nicht inline bereitstellen, so wird für die Konvertierung eine PC-spezifische Schrift aus den Angaben in der `mapping.def` abgeleitet.

Fehlen die Angaben in der `mapping.def`, so versucht AFP2web aus der Namenskonvention die nötigen Schriftinformationen abzuleiten. So ist AFP2web in der Lage, Ersatzschriften für AFP Schriften zu bestimmen.

In diesem Abschnitt beschreiben wir, nach welchen Regeln AFP2web aus dem Ressourcennamen die Kriterien für die Auswahl einer PC-Schrift ableitet. Beachten Sie bitte, dass wir die IBM spezifische Konvention zugrunde legen.

Die AFP Schriftnamenskonvention

Die Allgemeine Struktur

Der Name eines AFP Character Sets oder Coded Fonts enthält bis zu 8 Zeichen mit folgender Bedeutung:

A	F	R	S	T	C	P	X
----------	----------	----------	----------	----------	----------	----------	----------

	<i>Bedeutung</i>	<i>Von AFP2web verwendet?</i>
A	Komponente	Ja
F	Format oder Ausrichtung	Wird ignoriert
R	Schriftfamilie	Ja
S	Schriftschnitt	Ja
T	Schriftstärke	Ja
C	Subset zur Schriftfamilie	Wird ignoriert
P	Punktgröße	Ja
X	Wird für die Code Page Zuordnung verwendet	Wird ignoriert

Beispiel: COT07560 enthält als Name einer AFP Ressource die folgende Information:

A=C	Die Ressource ist ein Character Set
F=0	(Wird ignoriert)
R=T	Die Schriftfamilie ist Sonoran Serif
S=0	Der Schriftschnitt ist Roman
T=7	Schriftstärke Bold
C=5	(Wird ignoriert)
P=6	Die Punktgröße ist 6
X=0	(Wird ignoriert)

Die Codes werden in den folgenden Tabellen aufgeschlüsselt.

A: Komponente

A	Komponente
C	Character Set
X	Coded Font
A	Component

R: Schriftfamilie

Für "normale" Schrift ist R ein Wert aus:

R	Schriftfamilie	AFP2web Standard- Schriftfamilie für PDF	AFP2web Standard- Schriftfamilie für UNIX TIFF	AFP2web Standard- Schriftfamilie für Windows TIFF
4	Courier	Courier	courier	Courier
5	Letter Gothic	Courier	courier	Courier
6	Gothic Text	Courier	courier	Courier
7	Prestige	Courier	courier	Courier
8	Boldface	Times-Roman	times	Times New Roman
9	OCR	Courier	courier	Courier
B	BookMaster	Courier	courier	Courier
H	Helvetica	Helvetica	helvetica	Helvetica
N	Times New Roman	Times-Roman	times	Times New Roman

Für lizenzierte Fonts gilt für R einer der folgenden Werte:

R	Schriftfamilie	AFP2web Standard- Schriftfamilie für PDF	AFP2web Standard- Schriftfamilie für UNIX TIFF	AFP2web Standard- Schriftfamilie für Windows TIFF
2	Proprinter Emulation	Helvetica	helvetica	Helvetica

<i>R</i>	<i>Schriftfamilie</i>	<i>AFP2web Standard- Schriftfamilie für PDF</i>	<i>AFP2web Standard- Schriftfamilie für UNIX TIFF</i>	<i>AFP2web Standard- Schriftfamilie für Windows TIFF</i>
A	Sonoran Sans Serif	Helvetica	helvetica	Helvetica
B	Bar Code	Helvetica	helvetica	Helvetica
C	Century School- book	NewCentury- SchlBk	new century schoolbook	Century School- book
G	Monotype Garamond	Courier	courier	Courier
J	Sonoran Dis- play	Times-Roman	times	Times New Roman
M	Mathematic and Science	Symbol	symbol	Symbol
O	Optical Cha- racter Recogni- tion	Courier	courier	Courier
P	Pi Sans Serif	Helvetica	helvetica	Helvetica
Q	Pi Serif	Times-Roman	times	Times New Roman
S	ITC Souvenir	Helvetica	helvetica	Helvetica
T	Sonoran Serif	Times-Roman	times	Times New Roman
V	ITC Avant Garde Gothic	Helvetica	helvetica	Helvetica
Z	Sonoran Petite	Times-Roman	times	Times New Roman

S: Schriftschnitt

<i>S</i>	<i>Schriftschnitt</i>	<i>AFP2web Standardwert Schriftschnitt</i>
0	Roman	Medium
1	Italic	Italic
2	Roman Medium	Medium
3	Italic Medium	Italic
4	Roman Bold	Bold
5	Italic Bold	Italic Bold
6	Roman Medium Reverse	Medium

T: Schriftstärke

Für eine "normale" Schrift wird T ignoriert. Für eine lizenzierte Schrift ist T ein Wert aus

<i>T</i>	<i>Schriftstärke</i>	<i>AFP2web Standard- Stärke TIFF Output unter Windows</i>	<i>AFP2web Standard- Stärke: PDF und TIFF Output unter Unix</i>
1	Ultralight	Light	Medium
2	Extralight	Light	Medium
3	Light	Light	Medium
4	Semilight	Light	Medium
5	Medium	Medium	Medium
6	Semibold	Bold	Bold
7	Bold	Bold	Bold
8	Extrabold	Bold	Bold
9	Ultrabold	Bold	Bold

P: Punktgröße

<i>P</i>	<i>Punktgröße (points)</i>	<i>P</i>	<i>Punktgröße (points)</i>
4	4	K	21
5	5	L	22

<i>P</i>	<i>Punktgröße (points)</i>	<i>P</i>	<i>Punktgröße (points)</i>
6	6	M	23
7	7	N	24
8	8	O	25
9	9	P	26
0	10	Q	27
A	11	R	28
B	12	S	29
C	13	T	30
D	14	U	31
E	15	V	32
F	16	W	33
G	17	X	34
H	18	Y	35
I	19	Z	36
J	20		

5.8 Scripting Facility Schnell-Referenz

Scripting Facility Routinen und Packages

Die Scripting Facility bietet eine Schnittstelle für die Interaktion mit AFP2web Ereignissen. Ein Skript muss hierfür die folgenden Subroutinen definieren:

- sub afp2web()
- sub initialize()
- sub initializeDoc()
- sub initializePage()
- sub finalizePage()
- sub finalizeDoc()
- sub finalize()

Die folgenden Seiten geben einen Überblick über die API Methoden und die Subroutinen, in denen eine Methode genutzt werden kann.

Skript Routinen und Methoden

<i>Script Routine</i>	<i>Package</i>	<i>Methoden</i>
initialize	a2w::Config	getAttribute, getIndexFilePath, getOutputFilePath, getScriptArgs, setAttribute, setAutoSplit
	a2w::Font	getEncoding, getHeight, getName, getWidth, isBold, isItalic, isLight, isStrikeover, isUnderline, new, setBold, setEncoding, setFamilyName, setHeight, setItalic, setLight, setName, setStrikeover, setUnderline, setWidth
	a2w::Kernel	getIndexFilename, getResourceFilename, getSpoolFilename
	a2w::Page	new
initializeDoc	a2w::Config	getAttribute, getIndexFilePath, getOutputFilePath, getScriptArgs
	a2w::Document	addBookmark, addIndex, addNOP, addPage, getFirstIndex, getld, getName, getNextIndex, getOutputFilename, getPID, getSimpleFilename, setName, setOutputFilename
	a2w::Font	getEncoding, getHeight, getName, getWidth, isBold, isItalic, isLight, isStrikeover, isUnderline, new, setBold, setEncoding, setFamilyName, setHeight, setItalic, setLight, setName, setStrikeover, setUnderline, setWidth
	a2w::Index	getEBCDICName, getEBCDICValue, getIndexObjectName, getIndexObjectType, getLevelNumber, getName, getNameLength, getSequenceNumber, getValue, getValueLength, new, remove, setLevelNumber, setName, setSequenceNumber, setValue
	a2w::Kernel	getIndexFilename, getResourceFilename, getSpoolFilename
	a2w::NOP	getEBCDICValue, getLength, getValue, new, remove, setValue
	a2w::Page	new
initializePage	a2w::Config	getAttribute, getIndexFilePath, getOutputFilePath, getScriptArgs

<i>Script Routine</i>	<i>Package</i>	<i>Methoden</i>
	a2w::Document	addBookmark, addIndex, addNOP, addPage, getFirstIndex, getId, getName, getNextIndex, getOutputFilename, getPID, getSimpleFilename, setName, setOutputFilename
	a2w::Font	getEncoding, getHeight, getName, getWidth, isBold, isItalic, isLight, isStrikeover, isUnderline, new, setBold, setEncoding, setFamilyName, setHeight, setItalic, setLight, setName, setStrikeover, setUnderline, setWidth
	a2w::Index	getEBCDICName, getEBCDICValue, getIndexedObjectName, getIndexedObjectType, getLevelNumber, getName, getNameLength, getSequenceNumber, getValue, getValueLength, new, remove, setLevelNumber, setName, setSequenceNumber, setValue
	a2w::Kernel	getIndexFilename, getResourceFilename, getSpoolFilename
	a2w::Line	getColor, getLength, getWidth, getXPos, getYPos, isHorizontal, isNegative, isVertical, new, remove, setColor, setHorizontal, setLength, setNegative, setVertical, setWidth, setXPos, setYPos
	a2w::MediumMap	getDuplexControl, getFormdefName, getHeight, getName, getNupControl, getResolution, getWidth
	a2w::NOP	getEBCDICValue, getLength, getValue, new, remove, setValue
	a2w::Overlay	getFirstLine, getFirstOverlay, getFirstPageSegment, getFirstText, getHeight, getIncludedXPosition, getIncludedYPosition, getName, getNextLine, getNextOverlay, getNextPageSegment, getNextText, getResolution, getWidth
	a2w::PSEG	getIncludedXPosition, getIncludedYPosition, getName

<i>Script Routine</i>	<i>Package</i>	<i>Methoden</i>
	a2w::Page	addAnnotation, addBookmark, addImage, addIndex, addLine, addNOP, addOverlay, addPSEG, addText, getAppliedMediumMap, getFirstIndex, getFirstLine, getFirstMediumOverlay, getFirstNOP, getFirstOverlay, getFirstPageSegment, getFirstText, getHeight, getName, getNextIndex, getNextLine, getNextMediumOverlay, getNextNOP, getNextOverlay, getNextPageSegment, getNextText, getOutputFilename, getPageGroupName, getParsedId, getResolution, getSimpleFilename, getText, getType, getWidth, isConstantBack, isOverlayFound, new, setBackgroundColor, setHeight, setOutputFilename, setResolution, setWidth
	a2w::Text	getAngle, getColor, getEBCDICText, getFont, getMappedFontLocalId, getText, getTextLen, getXPos, getYPos, new, remove, setAngle, setColor, setFont, setText, setTextLen, setXPos, setYPos
afp2web	a2w::Config	getAttribute, getIndexFilePath, getOutputFilePath, getScriptArgs
	a2w::Document	addBookmark, addIndex, addNOP, addPage, getFirstIndex, getId, getName, getNextIndex, getOutputFilename, getPID, getSimpleFilename, setName, setOutputFilename
	a2w::Font	getCharacterSetName, getCodedFontName, getCodePageName, getEncoding, getHeight, getName, getTypefaceName, getWidth, isBold, isItalic, isLight, isStrikeover, isUnderline, new, setBold, setEncoding, setFamilyName, setHeight, setItalic, setLight, setName, setStrikeover, setUnderline, setWidth
	a2w::Index	getEBCDICName, getEBCDICValue, getIndexObjectName, getIndexObjectType, getLevelNumber, getName, getNameLength, getSequenceNumber, getValue, getValueLength, new, remove, setLevelNumber, setName, setSequenceNumber, setValue
	a2w::Kernel	getIndexFilename, getResourceFilename, getSpoolFilename

<i>Script Routine</i>	<i>Package</i>	<i>Methoden</i>
	a2w::Line	getColor, getLength, getWidth, getXPos, getYPos, isHorizontal, isNegative, isVertical, new, remove, setColor, setHorizontal, setLength, setNegative, setVertical, setWidth, setXPos, setYPos
	a2w::MediumMap	getDuplexControl, getFormdefName, getHeight, getName, getNupControl, getResolution, getWidth
	a2w::NOP	getEBCDICValue, getLength, getValue, new, remove, setValue
	a2w::Overlay	getFirstLine, getFirstOverlay, getFirstPageSegment, getFirstText, getHeight, getIncludedXPosition, getIncludedYPosition, getName, getNextLine, getNextOverlay, getNextPageSegment, getNextText, getResolution, getWidth
	a2w::PSEG	getIncludedXPosition, getIncludedYPosition, getName
	a2w::Page	addAnnotation, addBookmark, addImage, addIndex, addLine, addNOP, addOverlay, addPSEG, addText, getAppliedMediumMap, getFirstIndex, getFirstLine, getFirstMediumOverlay, getFirstNOP, getFirstOverlay, getFirstPageSegment, getFirstText, getHeight, getName, getNextIndex, getNextLine, getNextMediumOverlay, getNextNOP, getNextOverlay, getNextPageSegment, getNextText, getOutputFilename, getPageGroupName, getParsed, getResolution, getSimpleFilename, getText, getType, getWidth, isConstantBack, isOverlayFound, new, setBackgroundColor, setHeight, setOutputFilename, setResolution, setWidth
	a2w::Text	getAngle, getColor, getEBCDICText, getFont, getMappedFontLocalId, getText, getTextLen, getXPos, getYPos, new, remove, setAngle, setColor, setFont, setText, setTextLen, setXPos, setYPos
finalizePage	a2w::Config	getAttribute, getIndexFilePath, getOutputFilePath, getScriptArgs
	a2w::Document	addBookmark, addIndex, addNOP, addPage, getFirstIndex, getId, getName, getNextIndex, getOutputFilename, getPID, getSimpleFilename, setName, setOutputFilename

<i>Script Routine</i>	<i>Package</i>	<i>Methoden</i>
	a2w::Font	getCharacterSetName, getCodedFont-Name, getCodePageName, getEncoding, getHeight, getName, getTypefaceName, getWidth, isBold, isItalic, isLight, isStrikeover, isUnderline, new, setBold, setEncoding, setFamilyName, setHeight, setItalic, setLight, setName, setStrikeover, setUnderline, setWidth
	a2w::Index	getEBCDICName, getEBCDICValue, getIndexedObjectName, getIndexedObjectType, getLevelNumber, getName, getNameLength, getSequenceNumber, getValue, getValueLength, new, remove, setLevelNumber, setName, setSequenceNumber, setValue
	a2w::Kernel	getIndexFilename, getResourceFilename, getSpoolFilename
	a2w::Line	getColor, getLength, getWidth, getXPos, getYPos, isHorizontal, isNegative, isVertical, new, remove, setColor, setHorizontal, setLength, setNegative, setVertical, setWidth, setXPos, setYPos
	a2w::MediumMap	getDuplexControl, getFormdefName, getHeight, getName, getNupControl, getResolution, getWidth
	a2w::NOP	getEBCDICValue, getLength, getValue, new, remove, setValue
	a2w::Overlay	getFirstLine, getFirstOverlay, getFirstPageSegment, getFirstText, getHeight, getIncludedXPosition, getIncludedYPosition, getName, getNextLine, getNextOverlay, getNextPageSegment, getNextText, getResolution, getWidth
	a2w::PSEG	getIncludedXPosition, getIncludedYPosition, getName

<i>Script Routine</i>	<i>Package</i>	<i>Methoden</i>
	a2w::Page	addAnnotation, addBookmark, addImage, addIndex, addLine, addNOP, addOverlay, addPSEG, addText, getAppliedMediumMap, getFirstIndex, getFirstLine, getFirstMediumOverlay, getFirstNOP, getFirstOverlay, getFirstPageSegment, getFirstText, getHeight, getName, getNextIndex, getNextLine, getNextMediumOverlay, getNextNOP, getNextOverlay, getNextPageSegment, getNextText, getOutputFilename, getPageGroupName, getParsedId, getResolution, getSimpleFilename, getText, getType, getWidth, isConstantBack, isOverlayFound, new, setBackgroundColor, setHeight, setOutputFilename, setResolution, setWidth
	a2w::Text	getAngle, getColor, getEBCDICText, getFont, getMappedFontLocalId, getText, getTextLen, getXPos, getYPos, new, remove, setAngle, setColor, setFont, setText, setTextLen, setXPos, setYPos
finalizeDoc	a2w::Config	getAttribute, getIndexFilePath, getOutputFilePath, getScriptArgs
	a2w::Document	getFirstIndex, getFirstPage, getId, getName, getNextIndex, getNextPage, getOutputBuffer, getOutputBufferLength, getOutputFilename, getPageCount, getPID, getSimpleFilename, getSize
	a2w::Font	getCharacterSetName, getCodedFontName, getCodePageName, getEncoding, getHeight, getName, getTypefaceName, getWidth, isBold, isItalic, isLight, isStrikeover, isUnderline
	a2w::Index	getEBCDICName, getEBCDICValue, getIndexedObjectName, getIndexedObjectType, getLevelNumber, getName, getNameLength, getSequenceNumber, getValue, getValueLength
	a2w::Kernel	getIndexFilename, getResourceFilename, getSpoolFilename
	a2w::Line	getColor, getLength, getWidth, getXPos, getYPos, isHorizontal, isNegative, isVertical
	a2w::MediumMap	getDuplexControl, getFormdefName, getHeight, getName, getNupControl, getResolution, getWidth

<i>Script Routine</i>	<i>Package</i>	<i>Methoden</i>
	a2w::NOP	getEBCDICValue, getLength, getValue
	a2w::Overlay	getFirstLine, getFirstOverlay, getFirstPageSegment, getFirstText, getHeight, getIncludedXPosition, getIncludedYPosition, getName, getNextLine, getNextOverlay, getNextPageSegment, getNextText, getResolution, getWidth
	a2w::PSEG	getIncludedXPosition, getIncludedYPosition, getName
	a2w::Page	getAppliedMediumMap, getFirstIndex, getFirstLine, getFirstMediumOverlay, getFirstNOP, getFirstOverlay, getFirstPageSegment, getFirstText, getHeight, getId, getName, getNextIndex, getNextLine, getNextMediumOverlay, getNextNOP, getNextOverlay, getNextPageSegment, getNextText, getOutputBuffer, getOutputBufferLength, getOutputFilename, getPageGroupName, getParsedId, getResolution, getSimpleFilename, getSize, getText, getType, getWidth, isConstantBack, isOverlayFound
	a2w::Text	getAngle, getColor, getEBCDICText, getFont, getMappedFontLocalId, getText, getTextLen, getXPos, getYPos
finalize	a2w::Config	getAttribute, getIndexFilePath, getOutputFilePath, getScriptArgs
	a2w::Kernel	getIndexFilename, getResourceFilename, getSpoolFilename

Methoden und deren Verfügbarkeit

<i>a2w::Config Methoden</i>	<i>Verfügbar in den Skriptroutinen</i>
getAttribute	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc, finalize
getIndexFilePath	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc, finalize
getOutputFilePath	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc, finalize
getScriptArgs	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc, finalize
setAttribute	initialize
setAutoSplit	initialize

<i>a2w::Document Methoden</i>	<i>Verfügbar in den Skriptroutinen</i>
addBookmark	initializeDoc, initializePage, afp2web, finalizePage
addIndex	initializeDoc, initializePage, afp2web, finalizePage
addNOP	initializeDoc, initializePage, afp2web, finalizePage
addPage	initializeDoc, initializePage, afp2web, finalizePage
getFirstIndex	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getFirstPage	finalizeDoc
getId	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getName	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getNextIndex	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getNextPage	finalizeDoc
getOutputBuffer	finalizeDoc
getOutputBufferLength	finalizeDoc
getOutputFilename	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getPageCount	finalizeDoc

<i>a2w::Document Methoden</i>	<i>Verfügbar in den Skript Routinen</i>
getPID	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getSimpleFilename	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getSize	finalizeDoc
setName	initializeDoc, initializePage, afp2web, finalizePage
setOutputFilename	initializeDoc, initializePage, afp2web, finalizePage

<i>a2w::Font Methoden</i>	<i>Verfügbar in den Skript Routinen</i>
getCharacterSetName	afp2web, finalizePage, finalizeDoc
getCodedFontName	afp2web, finalizePage, finalizeDoc
getCodePageName	afp2web, finalizePage, finalizeDoc
getEncoding	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getHeight	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getName	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getTypefaceName	afp2web, finalizePage, finalizeDoc
getWidth	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
isBold	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
isItalic	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
isLight	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
isStrikeover	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
isUnderline	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
new	initialize, initializeDoc, initializePage, afp2web, finalizePage
setBold	initialize, initializeDoc, initializePage, afp2web, finalizePage
setEncoding	initialize, initializeDoc, initializePage, afp2web, finalizePage

<i>a2w::Font Methoden</i>	<i>Verfügbar in den Skriptroutinen</i>
setFamilyName	initialize, initializeDoc, initializePage, afp2web, finalizePage
setHeight	initialize, initializeDoc, initializePage, afp2web, finalizePage
setItalic	initialize, initializeDoc, initializePage, afp2web, finalizePage
setLight	initialize, initializeDoc, initializePage, afp2web, finalizePage
setName	initialize, initializeDoc, initializePage, afp2web, finalizePage
setStrikeover	initialize, initializeDoc, initializePage, afp2web, finalizePage
setUnderline	initialize, initializeDoc, initializePage, afp2web, finalizePage
setWidth	initialize, initializeDoc, initializePage, afp2web, finalizePage

<i>a2w::Index Methoden</i>	<i>Verfügbar in den Skriptroutinen</i>
getEBCDICName	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getEBCDICValue	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getIndexedObjectName	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getIndexedObjectType	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getLevelNumber	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getName	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getNameLength	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getSequenceNumber	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getValue	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getValueLength	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
new	initializeDoc, initializePage, afp2web, finalizePage

<i>a2w::Index Methoden</i>	<i>Verfügbar in den Skript Routinen</i>
remove	initializeDoc, initializePage, afp2web, finalizePage
setLevelNumber	initializeDoc, initializePage, afp2web, finalizePage
setName	initializeDoc, initializePage, afp2web, finalizePage
setSequenceNumber	initializeDoc, initializePage, afp2web, finalizePage
setValue	initializeDoc, initializePage, afp2web, finalizePage

<i>a2w::Kernel Methoden</i>	<i>Verfügbar in den Skript Routinen</i>
getIndexFilename	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc, finalize
getResourceFilename	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc, finalize
getSpoolFilename	initialize, initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc, finalize

<i>a2w::Line Methoden</i>	<i>Verfügbar in den Skript Routinen</i>
getColor	initializePage, afp2web, finalizePage, finalizeDoc
getLength	initializePage, afp2web, finalizePage, finalizeDoc
getWidth	initializePage, afp2web, finalizePage, finalizeDoc
getXPos	initializePage, afp2web, finalizePage, finalizeDoc
getYPos	initializePage, afp2web, finalizePage, finalizeDoc
isHorizontal	initializePage, afp2web, finalizePage, finalizeDoc
isNegative	initializePage, afp2web, finalizePage, finalizeDoc
isVertical	initializePage, afp2web, finalizePage, finalizeDoc
new	initializePage, afp2web, finalizePage
remove	initializePage, afp2web, finalizePage
setColor	initializePage, afp2web, finalizePage
setHorizontal	initializePage, afp2web, finalizePage
setLength	initializePage, afp2web, finalizePage
setNegative	initializePage, afp2web, finalizePage
setVertical	initializePage, afp2web, finalizePage
setWidth	initializePage, afp2web, finalizePage
setXPos	initializePage, afp2web, finalizePage
setYPos	initializePage, afp2web, finalizePage

<i>a2w::MediumMap Methoden</i>	<i>Verfügbar in den Skript Routinen</i>
getDuplexControl	initializePage, afp2web, finalizePage, finalizeDoc
getFormdefName	initializePage, afp2web, finalizePage, finalizeDoc
getHeight	initializePage, afp2web, finalizePage, finalizeDoc
getName	initializePage, afp2web, finalizePage, finalizeDoc
getNupControl	initializePage, afp2web, finalizePage, finalizeDoc
getResolution	initializePage, afp2web, finalizePage, finalizeDoc
getWidth	initializePage, afp2web, finalizePage, finalizeDoc

<i>a2w::NOP Methoden</i>	<i>Verfügbar in den Skript Routinen</i>
getEBCDICValue	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getLength	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
getValue	initializeDoc, initializePage, afp2web, finalizePage, finalizeDoc
new	initializeDoc, initializePage, afp2web, finalizePage
remove	initializeDoc, initializePage, afp2web, finalizePage
setValue	initializeDoc, initializePage, afp2web, finalizePage

<i>a2w::Overlay Methoden</i>	<i>Verfügbar in den Skript Routinen</i>
getFirstLine	initializePage, afp2web, finalizePage, finalizeDoc
getFirstOverlay	initializePage, afp2web, finalizePage, finalizeDoc
getFirstPageSegment	initializePage, afp2web, finalizePage, finalizeDoc
getFirstText	initializePage, afp2web, finalizePage, finalizeDoc
getHeight	initializePage, afp2web, finalizePage, finalizeDoc
getIncludedXPosition	initializePage, afp2web, finalizePage, finalizeDoc
getIncludedYPosition	initializePage, afp2web, finalizePage, finalizeDoc
getName	initializePage, afp2web, finalizePage, finalizeDoc
getNextLine	initializePage, afp2web, finalizePage, finalizeDoc
getNextOverlay	initializePage, afp2web, finalizePage, finalizeDoc
getNextPageSegment	initializePage, afp2web, finalizePage, finalizeDoc
getNextText	initializePage, afp2web, finalizePage, finalizeDoc
getResolution	initializePage, afp2web, finalizePage, finalizeDoc

<i>a2w::Overlay Methoden</i>	<i>Verfügbar in den Skript Routinen</i>
getWidth	initializePage, afp2web, finalizePage, finalizeDoc

<i>a2w::PSEG Methoden</i>	<i>Verfügbar in den Skript Routinen</i>
getIncludedXPosition	initializePage, afp2web, finalizePage, finalizeDoc
getIncludedYPosition	initializePage, afp2web, finalizePage, finalizeDoc
getName	initializePage, afp2web, finalizePage, finalizeDoc

<i>a2w::Page Methoden</i>	<i>Verfügbar in den Skript Routinen</i>
addAnnotation	initializePage, afp2web, finalizePage
addBookmark	initializePage, afp2web, finalizePage
addImage	initializePage, afp2web, finalizePage
addIndex	initializePage, afp2web, finalizePage
addLine	initializePage, afp2web, finalizePage
addNOP	initializePage, afp2web, finalizePage
addOverlay	initializePage, afp2web, finalizePage
addPSEG	initializePage, afp2web, finalizePage
addText	initializePage, afp2web, finalizePage
getAppliedMediumMap	initializePage, afp2web, finalizePage, finalizeDoc
getFirstIndex	initializePage, afp2web, finalizePage, finalizeDoc
getFirstLine	initializePage, afp2web, finalizePage, finalizeDoc
getFirstMediumOverlay	initializePage, afp2web, finalizePage, finalizeDoc
getFirstNOP	initializePage, afp2web, finalizePage, finalizeDoc
getFirstOverlay	initializePage, afp2web, finalizePage, finalizeDoc
getFirstPageSegment	initializePage, afp2web, finalizePage, finalizeDoc
getFirstText	initializePage, afp2web, finalizePage, finalizeDoc
getHeight	initializePage, afp2web, finalizePage, finalizeDoc
getId	finalizeDoc
getName	initializePage, afp2web, finalizePage, finalizeDoc
getNextIndex	initializePage, afp2web, finalizePage, finalizeDoc
getNextLine	initializePage, afp2web, finalizePage, finalizeDoc
getNextMediumOverlay	initializePage, afp2web, finalizePage, finalizeDoc
getNextNOP	initializePage, afp2web, finalizePage, finalizeDoc

<i>a2w::Page Methoden</i>	<i>Verfügbar in den Skript Routinen</i>
getNextOverlay	initializePage, afp2web, finalizePage, finalizeDoc
getNextPageSegment	initializePage, afp2web, finalizePage, finalizeDoc
getNextText	initializePage, afp2web, finalizePage, finalizeDoc
getOutputBuffer	finalizeDoc
getOutputBufferLength	finalizeDoc
getOutputFilename	initializePage, afp2web, finalizePage, finalizeDoc
getPageGroupName	initializePage, afp2web, finalizePage, finalizeDoc
getParseId	initializePage, afp2web, finalizePage, finalizeDoc
getResolution	initializePage, afp2web, finalizePage, finalizeDoc
getSimpleFilename	initializePage, afp2web, finalizePage, finalizeDoc
getSize	finalizeDoc
getText	initializePage, afp2web, finalizePage, finalizeDoc
getType	initializePage, afp2web, finalizePage, finalizeDoc
getWidth	initializePage, afp2web, finalizePage, finalizeDoc
isConstantBack	initializePage, afp2web, finalizePage, finalizeDoc
isOverlayFound	initializePage, afp2web, finalizePage, finalizeDoc
new	initialize, initializeDoc, initializePage, afp2web, finalizePage
setBackgroundColor	initializePage, afp2web, finalizePage
setHeight	initializePage, afp2web, finalizePage
setOutputFilename	initializePage, afp2web, finalizePage
setResolution	initializePage, afp2web, finalizePage
setWidth	initializePage, afp2web, finalizePage

<i>a2w::Text Methoden</i>	<i>Verfügbar in den Skript Routinen</i>
getAngle	initializePage, afp2web, finalizePage, finalizeDoc
getColor	initializePage, afp2web, finalizePage, finalizeDoc
getEBCDICText	initializePage, afp2web, finalizePage, finalizeDoc
getFont	initializePage, afp2web, finalizePage, finalizeDoc
getMappedFontLocalId	initializePage, afp2web, finalizePage, finalizeDoc
getText	initializePage, afp2web, finalizePage, finalizeDoc
getTextLen	initializePage, afp2web, finalizePage, finalizeDoc

<i>a2w::Text Methoden</i>	<i>Verfügbar in den Skript Routinen</i>
getXPos	initializePage, afp2web, finalizePage, finalizeDoc
getYPos	initializePage, afp2web, finalizePage, finalizeDoc
new	initializePage, afp2web, finalizePage
remove	initializePage, afp2web, finalizePage
setAngle	initializePage, afp2web, finalizePage
setColor	initializePage, afp2web, finalizePage
setFont	initializePage, afp2web, finalizePage
setText	initializePage, afp2web, finalizePage
setTextLen	initializePage, afp2web, finalizePage
setXPos	initializePage, afp2web, finalizePage
setYPos	initializePage, afp2web, finalizePage

5.9 Scripting Facility API Reference

Overview

Conventions used in this document

- “[NEW]” indicates a new module or interface.
- “[DEPRECATED]” indicates that a module or interface, which is no longer recommended. It will be removed in the future. The same functionality can be achieved using another new generic interface.
- “[MODIFIED]” indicates a change to an existing module or interface.
- “[EXTENDED]” indicates additional functionality.

afp2web.pm

The module afp2web.pm is the gateway script of the AFP2web Scripting Facility and this module will be loaded and executed by the AFP2web Kernel (using an embedded PERL interpreter). The description below lists the interfaces exposed by the module afp2web.pm and which can be used to communicate with the AFP2web Kernel.

afp2web

Parameter: None

Return value data type: Integer

Remarks: Called while processing (after binding resources) each output page. At this stage, it is possible to analyse page content and, by setting the appropriate return code, to instruct the AFP2web Kernel to change its flow. One of the following return values are possible:

Return value	Description
<0	A negative value indicates an error to AFP2web. When returning an error code to AFP2web, it is also possible to pass an error message for a more descriptive error report.
0	Append page to current document
1	Skip page
2	First page / New document

Example:

```
sub afp2web(){
$APPEND = 0;# append page to Current Document
$SKIP = 1;# skip page
$NEWDOC = 2;# new document
$svRetTmp = $APPEND; # default: append page
#---- Do something
...
#---- Return value
if ( $svProcessOk ){
return $svRetTmp;#---- Success
} else {
return (-1, "afp2web failed" );#---- Error
}
}
```

finalize

Parameter: None

Return value Integer
data type:

Remarks: Called after processing each spool. This routine is used for spool finalizing.

The return value must be zero or greater than zero. A negative value indicates an error to AFP2web. When returning an error code to AFP2web, it is also possible to pass an error message for a more descriptive error report.

Example:

```
sub finalize(){
#---- Do something
...
#---- Return value
if ( $svProcessOk ){
return 0;#---- Success
} else {
return (-1, "afp2web failed" );#---- Error
}
}
```

finalizeDoc

Parameter: None

**Return value
data type:** Integer

Remarks: Called after processing each output document. This routine is used for document finalizing.

The return value must be zero or greater than zero. A negative value indicates an error to AFP2web. When returning an error code to AFP2web, it is also possible to pass an error message for a more descriptive error report.

Example:

```
sub finalizeDoc(){
#---- Do something
...
#---- Return value
if ( $svProcessOk ){
return 0; #---- Success
} else {
return (-1, "afp2web failed" ); #---- Error
}
}
```

finalizePage

Parameter: None

**Return value
data type:** Integer

Remarks: Called after processing each output page. This routine is used for page finalizing.

The return value must be zero or greater than zero. A negative value indicates an error to AFP2web. When returning an error code to AFP2web, it is also possible to pass an error message for a more descriptive error report.

Example:

```
sub finalizePage(){
#---- Do something
...
#---- Return value
if ( $svProcessOk ){
return 0; #---- Success
} else {
return (-1, "afp2web failed" ); #---- Error
}
}
```

initialize

Parameter: Configuration instance of type a2w::Config
Kernel instance of type a2w::Kernel

Return value Integer
data type:

Remarks: Called at the beginning of each input spool. Using a configuration instance, it is possible to read and modify configuration parameters (which were entered in INI file or as command line options). Using an instance of the kernel object, it is possible to read information about the input spool.

The return value must be zero or greater than zero. A negative value indicates an error to AFP2web. When returning an error code to AFP2web, it is also possible to pass an error message for a more descriptive error report.

Example:

```
sub initialize(){
#---- Get parameters of initialize ( a2w::Config,
a2w::Kernel )
( $a2wConfigPar, $a2wKernelPar ) = @_ ;
#---- Do something
...
#---- Return value
if ( $svProcessOk ){
return 0; #---- Success
} else {
return (-1, "Initialization failed" ); #---- Error
}
}
```

initializeDoc

Parameter: Document instance of type a2w::Document

Return value Integer
data type:

Remarks: Called at the beginning of each output document process, the user can initialize output document dependent processes.

The return value must be zero or greater than zero. A negative value indicates an error to AFP2web. When returning an error code to AFP2web, it is also possible to pass an error message for a more descriptive error report.

Example:

```
sub initializeDoc(){
#---- Get parameters of initializeDoc ( a2w::Document )
( $a2wDocumentPar ) = @_ ;
#---- Do something
...
#---- Return value
if ( $svProcessOk ){
return 0; #---- Success
} else {
return (-1, "InitializeDoc failed" ); #---- Error
}
}
```

initializePage

Parameter: Page instance of type a2w::Page

Return value Integer
data type:

Remarks: Called at the beginning of each output page process, user can initialize output page dependent processes.

The return value must be zero or greater than zero. A negative value indicates an error to AFP2web. When returning an error code to AFP2web, it is also possible to pass an error message for a more descriptive error report.

Example:

```
sub initializePage(){
#---- Get parameters of initializePage ( a2w::Page )
( $a2wPagePar ) = @_;
#---- Do something
...
#---- Return value
if ( $svProcessOk ){
return 0;#---- Success
} else {
return (-1, "InitializePage failed" );#---- Error
}
}
```

a2w::Config (mhtIni [DEPRECATED])

The a2w::Config object wraps the AFP2web Kernel object mhtIni and exposes the following interfaces to the AFP2web Scripting Facility.

getAttribute [NEW]

Parameter: Attribute name of type string

**Return value
data type:** String

Remarks: Gets the current value of the INI attribute.

Example:

```
$svIndexPathTmp = $a2wConfig->getAttribute(
    "IndexPath" );
$svOutputFilePathTmp = $a2wConfig->getAttribute(
    "OutputFilePath" );
$svScriptArgsTmp = $a2wConfig->getAttribute( "ScriptArgument" );
```

getIndexFilePath [DEPRECATED]

Parameter: None

**Return value
data type:** String

Remarks: Gets the path specified for the index file.
Note: This method is deprecated. Please use the method "getAttribute".

Example:

```
$svIndexPathTmp = $a2wConfig->getIndexFilePath();
```

getOutputFilePath [DEPRECATED]

Parameter: None

**Return value
data type:** String

Remarks: Gets the path specified for the output file.
Note: This method is deprecated. Please use the method "getAttribute".

Example: `$svOutputFilePathTmp = $a2wConfig->getOutputFilePath();`

getScriptArgs [DEPRECATED]

Parameter: None

**Return value
data type:** String

Remarks: Gets the arguments passed to the script.
Note: This method is deprecated. Please use the method "getAttribute".

Example: `$svScriptArgsTmp = $a2wConfig->getScriptArgs();`

setAttribute [NEW]

Parameter: Attribute name of type string
Attribute value of type string

**Return value
data type:** Void

Remarks: Sets the INI attribute (the processing parameter, which is originally set in the configuration file called the INI file).

Example: `$a2wConfig->setAttribute("Quietmode", "on");`

setAutoSplit [DEPRECATED]

Parameter: Autosplit of type boolean

Return value Void
data type:

Remarks: Setting Autosplit to true means that document splitting will be done based on the standard AFP index elements. In this case, a return value of 2 of the `afp2web()` function of the AFP2web Scripting Facility module is ignored.

Setting Autosplit to false means that document splitting will be defined by the `afp2web()` function.

Note: This method is deprecated. Please use the method "setAttribute" with the key "AutoSplit" to set AutoSplit to "on" or "off".

Example: `$a2wConfig->setAutoSplit(1);`

a2w::Document (mhtDocument [DEPRECATED])

a2w::Document object wraps the AFP2web Kernal object “mhtDocument” and exposes the following interfaces to the Scripting Facility.

addBookmark [NEW]

Parameter: Bookmark name of type string
 Bookmark value of type string

Return value Void
data type:

Remarks: Adds a bookmark to the document.

Example: \$a2wDocument->addBookmark("Software", "AFP2web");

addIndex [DEPRECATED]

Parameter: Index name of type string
 Index value of type string

Return value Void
data type:

Remarks: Adds an index element to the document.
 This method is deprecated. Please use the method “addIndex” of this object.

Example: \$a2wDocument->addIndex("Producer", "Maas");

addIndex [NEW]

Parameter: Index instance of type a2w::Index.

Return value Void
data type:

Remarks: Adds an index element to the document.

Example:

```
#---- Create new index
$a2wIndexTmp = new a2w::Index();
#---- Set index info
$a2wIndexTmp->setName( "Producer" );
$a2wIndexTmp->setValue( "Maas" );
#---- Add index to document
$a2wDocument->addIndex( $a2wIndexTmp );
```

addNOP [NEW]

Parameter: NOP instance of type a2w::NOP

Return value Void
data type:

Remarks: Adds an NOP element to the document. The NOP element is inserted before first BPG of the document.

Example:

```
#---- Create new NOP
$a2wNOPTmp = new a2w::NOP();
#---- Set NOP value
$a2wNOPTmp->setValue( "Created by AFP2web" );
#---- Add NOP to document
$a2wDocument->addNOP( $a2wNOPTmp );
```

addPage [NEW]

Parameter: Page instance of type a2w::Page
Page identifier of the output document of type unsigned integer

Return value Void
data type:

Remarks: Adds a page to the document.

Example:

```
#---- Create new Page
$a2wPageTmp = new a2w::Page();
#---- Add some content
$a2wPageTmp->addText( $a2wTextTmp );
#---- Add Page to document
$a2wDocument->addPage( $a2wPageTmp, 2 );
```

getFirstIndex

Parameter: None

Return value Index instance of type a2w::Index
data type:

Remarks: Gets the first index at the document level.

Example:

```
$a2wIndexTmp = $a2wDocument->getFirstIndex();
while ( $a2wIndexTmp != 0 ){
#---- Access index info
...
#---- Get next index
$a2wIndexTmp = $a2wDocument->getNextIndex();
}
```

getFirstPage

Parameter: None

Return value Page instance of type a2w::Page
data type:

Remarks: Gets the first page of the document.

Example:

```
$a2wPageTmp = $a2wDocument->getFirstPage();  
while ($a2wPageTmp != 0 ){  
#---- Access page info  
...  
#---- Get next page  
$a2wPageTmp = $a2wDocument->getNextPage();  
}
```

getId

Parameter: None

**Return value
data type:** String

Remarks: Gets the document ID.

Example: \$svDocIdTmp = \$a2wDocument->getId();

getName

Parameter: None

**Return value
data type:** String

Remarks: Gets the document name.

Example: \$svDocNameTmp = \$a2wDocument->getName();

getNextIndex

Parameter: None

Return value Index instance of type *a2w::Index*
data type:

Remarks: Gets the next index at the document level. This method is normally used in an iteration loop after first using *getFirstIndex*.

Example:

```
$a2wIndexTmp = $a2wDocument->getFirstIndex();  
while ($a2wIndexTmp != 0 ){  
#---- Access index info  
...  
#---- Get next index  
$a2wIndexTmp = $a2wDocument->getNextIndex();  
}
```

getNextPage

Parameter: None

Return value Page instance of type *a2w::Page*
data type:

Remarks: Gets the next page of the document. This method is normally used in an iteration loop after first using *getFirstPage*.

Example:

```
$a2wPageTmp = $a2wDocument->getFirstPage();  
while ($a2wPageTmp != 0 ){  
#---- Access page info  
...  
#---- Get next page  
$a2wPageTmp = $a2wDocument->getNextPage();  
}
```

getOutputBuffer

Parameter: None

Return value Buffer of type *byte**
data type:

Remarks: Gets the document's output buffer.

Example: `$svDocBufferTmp = $a2wDocument->getOutputBuffer();`

getOutputBufferLength

Parameter: None

**Return value
data type:** Integer

Remarks: Gets the length of the document's output buffer.

Example: `$svDocBufferLenTmp = $a2wDocument->getOutputBufferLength();`

getOutputFilename

Parameter: None

**Return value
data type:** String

Remarks: Gets the name of the output file.

Example: `$svOutputFilenameTmp = $a2wDocument->getOutputFilename();`

getPID

Parameter: None

**Return value
data type:** String

Remarks: Gets the process ID.

Example: `$svProcessIdTmp = $a2wDocument->getPID();`

getPageCount

Parameter: None

**Return value
data type:** Integer

Remarks: Gets the document's page count.

Example: `$svPageCountTmp = $a2wDocument->getPageCount();`

getSimpleFilename

Parameter: None

**Return value
data type:** String

Remarks: Gets the simple file name of the output file.

Example: `$svSimpleFilenameTmp = $a2wDocument->getSimpleFilename();`

getSize

Parameter: None

**Return value
data type:** Long

Remarks: Gets the size of the output document.

Example: `$svDocSizeTmp = $a2wDocument->getSize();`

setName

Parameter: Document name of type string

**Return value
data type:** Void

Remarks: Sets the name of the document.

Example: `$a2wDocument->setName("MaasSample");`

setOutputFileName

Parameter: Output filename of type string

**Return value
data type:** Void

Remarks: Sets the name of the output file for this document.

Example: `$a2wDocument->setOutputFileName ("MaasSample.pdf");`

a2w::Font (*mhtAFPFont* [DEPRECATED])

The *a2w::Font* object wraps the AFP2web Kernel object *mhtAFPFont* and exposes the following interfaces to the AFP2web Scripting Facility.

getCharacterSetName [NEW]

Parameter: None

**Return value
data type:** String

Remarks: Gets the name of the character set.

Example: `$svCharSetNameTmp = $a2wFont->getCharacterSetName();`

getCodedFontName [NEW]

Parameter: None

**Return value
data type:** String

Remarks: Gets the coded font name.

Example: `$svCodedFontNameTmp = $a2wFont->getCodedFontName();`

getCodepageName [NEW]

Parameter: None

Return value String
data type:

Remarks: Gets the name of the mapped code page.

Example: `$svCodePageNameTmp = $a2wFont->getCodePageName();`

getEncoding [NEW]

Parameter: None

Return value String
data type:

Remarks: Returns the font encoding.

Example: `$svFontEncodingTmp = $a2wFont->getEncoding();`

getHeight [NEW]

Parameter: None

Return value Float
data type:

Remarks: Returns the font's height.

Example: `$svFontHeightTmp = $a2wFont->getHeight();`

getName [NEW]

Parameter: None

Return value String
data type:

Remarks: Returns the font name.

Example: \$a2wNameTmp = \$a2wFont->getName();

getTypefaceName [NEW]

Parameter: None

Return value String
data type:

Remarks: Returns the AFP font type face name.

Example: \$svTypefaceNameTmp = \$a2wFont->getTypefaceName();

getWidth [NEW]

Parameter: None

Return value Integer
data type:

Remarks: Returns the font's width.

Example: \$svFontWidthTmp = \$a2wFont->getWidth();

isBold [NEW]

Parameter: None

Return value Boolean
data type:

Remarks: Returns whether the font's weight is bold or not.

Example: `$svWeightTmp = $a2wFont->isBold();`

isItalic [NEW]

Parameter: None

Return value Boolean
data type:

Remarks: Returns whether the font's slant is italic or not.

Example: `$svSlantTmp = $a2wFont->isItalic();`

isLight [NEW]

Parameter: None

Return value Boolean
data type:

Remarks: Returns whether the font's weight is light or not.

Example: `$svWeightTmp = $a2wFont->isLight();`

isStrikeover [NEW]

Parameter: None

Return value Boolean
data type:

Remarks: Returns whether font is strike over or not.

Example: `$svStrikeOverTmp = $a2wFont->isStrikeover();`

isUnderline [NEW]

Parameter: None

Return value Boolean
data type:

Remarks: Returns whether font is underlined or not

Example: `$svUnderLineTmp = $a2wFont->isUnderLine();`

new

Parameter: None

Return value Font instance of type *a2w::Font*
data type:

Remarks: Allocates and returns a new font instance.

Example: `$a2wFont = new a2w::Font();`

setBold

Parameter: Font weight of type boolean

Return value Void
data type:

Remarks: Sets the font's weight to bold.

Example: `$a2wFont->setBold(1);`

setEncoding [NEW]

Parameter: Encoding of type string. Possible values are either ANSI, SYMBOL or NULL.

Return value Void
data type:

Remarks: Sets the encoding of the font.

Example: `$a2wFont->setEncoding("ANSI");`

setFamilyName

Parameter: Family name of type string

Return value Void
data type:

Remarks: Sets the family name of the font.

Example: `$a2wFont->setFamilyName("Latin1");`

setHeight

Parameter: Font height of type float

Return value Void
data type:

Remarks: Sets the height of the font. The value must be in points.
NOTE: On floating height value, only one digit is considered in the precision part. For example, the value "10.768" will be treated as "10.7".

Example: `$a2wFont->setHeight(100);`

setItalic

Parameter: Font slant of type boolean

Return value Void
data type:

Remarks: Sets the font's slant to italic

Example: `$a2wFont->setItalic(1);`

setLight

Parameter: Font weight of type boolean

Return value Void
data type:

Remarks: Sets the font's weight to light.

Example: `$a2wFont->setLight(1);`

setName

Parameter: Font name of type string

**Return value
data type:** Void

**Remarks for
PDF Output:** Sets the name of a font.
Font name should be one of the standard Type1 fonts as given below.

Times-Roman

Times-Bold

Times-Italic

Times-BoldItalic

Helvetica

Helvetica-Bold

Helvetica-Oblique

Helvetica-BoldOblique

Courier

Courier-Bold

Courier-Oblique

Courier-BoldOblique

Symbol

ZapfDingbats

The default font name is "Helvetica".

Note: Adobe has its own definition for above set of fonts, so they can be used with PDF.

**Example for
PDF Output:** `$a2wFont->setName("Times-Roman");`

**Remarks For
TIFF Output:** Font name is the typeface name of font. To find out the typeface name of a font follow given steps below appropriate to OS. Default font name is "Helvetica" for both Unix & Windows.

**Example for
TIFF Output:** `$a2wFont->setName("Verdana");`

**How to
Determine
Font Names
on Win-
dows:**

- Open "Control Panel->Fonts"
- Double click on the required font
- The font window will be displayed showing font information
- For TrueType fonts, use the name displayed for "Typeface name" as parameter to the Scripting Facility method "Font::setName()"
- For Type1 fonts, use the name given on top of the font window as parameter to Scripting Facility method "Font::setName()".

**How to
Determine
Font Names
on a Unix
Operating
System:**

- Type "xlsfonts" at the command prompt
- prompt\$>xlsfonts
- If issuing this command on the terminal window, then make sure that the "DISPLAY" environment variable is set properly with display ID.
 - The "xlsfonts" command will display the names of all available in XLFD (X Logical Font Descriptor) format (as given below):

prompt\$>xlsfonts

-adobe-courier-bold-o-normal--10-100-75-75-m-60-iso10646-1

-adobe-courier-medium-o-normal--10-100-75-75-m-60-iso10646-1

-adobe-helvetica-bold-o-normal--10-100-75-75-p-60-iso10646-1

-adobe-helvetica-medium-o-normal--10-100-75-75-p-57-iso10646-1

- In XLFD format, font attributes are separated by '-' character
- Use the second attribute that passes the typeface name of the font as parameter to setName().

**Example of
a Font
Name on
Unix:**

-adobe-courier-bold-o-normal--10-100-75-75-m-60-iso10646-1

Note: The text "courier" is the font typeface name.

`$a2wFont->setName("courier");`

setStrikeover

Parameter: Font strikeover of type boolean

**Return value
data type:** Void

Remarks: Sets the font to strike over.

Example: `$a2wFont->setStrikeover(1);`

setUnderline

Parameter: Font underline of type boolean

**Return value
data type:** Void

Remarks: Sets the font to underlined.

Example: `$a2wFont->setUnderline(1);`

setWidth

Parameter: Font width of type integer

**Return value
data type:** Void

Remarks: Sets the width of the font.

Example: `$a2wFont->setWidth(40);`

a2w::Index (mhtAttributeElement, mhtIndexElement, mhtPagePageGroupIndex [DEPRECATED])

The a2w::Index object wraps the AFP2web Kernel object mhtAttributeElement and exposes the following interfaces to the AFP2web Scripting Facility.

getEBCDICName [NEW]

Parameter: None

Return value data type: String

Remarks: Gets the the EBCDIC name of the index.

Example: `$svEBCDICNameTmp = $a2wIndex->getEBCDICName();`

getEBCDICValue [NEW]

Parameter: None

Return value data type: String

Remarks: Gets the index value in EBCDIC.

Example: `$svEBCDICValueTmp = $a2wIndex->getEBCDICValue();`

getIndexedObjectName

Parameter: None

**Return value
data type:** String

Remarks: Gets the name of the indexed object. The indexed object can be an indexed page or page group.

Example: `$svIndexedObjectNameTmp = $a2wIndex->getIndexedObjectName();`

getIndexedObjectType

Parameter: None

**Return value
data type:** Integer

Remarks: Gets the type of the indexed object (page or page group).

Example: `$svIndexedObjectTypeTmp = $a2wIndex->getIndexedObjectType();`

getLevelNumber

Parameter: None

**Return value
data type:** Integer

Remarks: Gets the level number (used to position hierarchically).

Example: `$svLevelNrTmp = $a2wIndex->getLevelNumber();`

getName [MODIFIED]

Parameter: None

**Return value
data type:** String

Remarks: Gets the index name.
Note: This method was previously named "getAttributeName".

Example: `$svNameTmp = $a2wIndex->getName();`

getNameLength [NEW]

Parameter: None

**Return value
data type:** Integer

Remarks: Gets the the length of the index name. This length is used to interpret the name which is retrieved as EBCDIC string.

Example: `$svNameLengthTmp = $a2wIndex->getNameLength();`

getSequenceNumber

Parameter: None

**Return value
data type:** Integer

Remarks: Gets the sequence number (used to identify identical indexes).

Example: `$svSeqNrTmp = $a2wIndex->getSequenceNumber();`

getValue [MODIFIED]

Parameter: None

**Return value
data type:** String

Remarks: Gets the index value
Note: This method was previously named "getAttributeValue".

Example: `$svValueTmp = $a2wIndex->getValue();`

getValueLength [NEW]

Parameter: None

**Return value
data type:** Integer

Remarks: Get the length of the index value in EBCDIC.

Example: `$svValueLengthTmp = $a2wIndex->getValueLength();`

new [NEW]

Parameter: None

**Return value
data type:** Index instance of type `a2w::Index`

Remarks: Allocates and returns the index instance.

Example: `$a2wIndexTmp = new a2w::Index();`

remove [NEW]

Parameter: None

Return value None
data type:

Remarks: Removes the index from the presentation data.

Example: `$a2wIndex->remove();`

setLevelNumber

Parameter: Level number of type integer

Return value Void
data type:

Remarks: Sets the level number (used to position hierarchically).

Example: `$a2wIndex->setLevelNumber(1);`

setName [MODIFIED]

Parameter: Index name of type string

Return value Void
data type:

Remarks: Sets the index name.
 Note: This method was previously called "setAttributeName".

Example: `$a2wIndex->setName("Producer");`

setSequenceNumber

Parameter: Sequence number of type integer

**Return value
data type:** Void

Remarks: Sets the sequence number of the index (used to identify identical indexes).

Example: `$a2wIndex->setSequenceNumber(10);`

setValue [MODIFIED]

Parameter: Index value of type string.

**Return value
data type:** Void

Remarks: Sets the index value.
Note: This method was previously called as “setAttributeValue”.

Example: `$a2wIndex->setValue("Maas");`

a2w::Kernel [NEW]

The a2w::Kernel object wraps the AFP2web Kernel object mhtAFP2web and exposes the following interfaces to the AFP2web Scripting Facility.

getIndexFilename [NEW]

Parameter: None

**Return value
data type:** String

Remarks: Returns name of the index file used for the current spool.

Example: `$svIndexFilenameTmp = $a2wKernel ->getIndexFilename();`

getResourceFilename [NEW]

Parameter: None

**Return value
data type:** String

Remarks: Returns the name of the resource file used for the current spool.

Example: `$svResFilenameTmp = $a2wKernel ->getResourceFilename();`

getSpoolFilename [NEW]

Parameter: None

Return value String
data type:

Remarks: Returns the name of the spool file, which is currently being processed.

Example: `$svSpoolFilenameTmp = $a2wKernel ->getSpoolFilename();`

a2w::Line [NEW]

The “a2w::Line” object wraps the AFP2web Kernel object mhtLineObject and exposes the following interfaces to the AFP2web Scripting Facility.

getColor [NEW]

Parameter: None

**Return value
data type:** Integer

Remarks: Returns the color of the line.

Example: `$svColorTmp = $a2wLine->getColor();`

getLength [NEW]

Parameter: None

**Return value
data type:** Integer

Remarks: Returns the line length.

Example: `$svLengthTmp = $a2wLine->getLength();`

getWidth [NEW]

Parameter: None

Return value Integer
data type:

Remarks: Returns the line width.

Example: `$svWidthTmp = $a2wLine->getWidth();`

getXPos [NEW]

Parameter: None

Return value Integer
data type:

Remarks: Returns the X position of the line.

Example: `$svXPosTmp = $a2wLine->getXPos();`

getYPos [NEW]

Parameter: None

Return value Integer
data type:

Remarks: Returns the Y position of the line.

Example: `$svYPosTmp = $a2wLine->getYPos();`

isHorizontal [NEW]

Parameter: None

**Return value
data type:** Boolean

Remarks: Returns whether the line is horizontal or not.

Example: `$svHorizontalTmp = $a2wLine->isHorizontal();`

isNegative [NEW]

Parameter: None

**Return value
data type:** Boolean

Remarks: Returns the line is set to negative or not.

Example: `$svNegativeTmp = $a2wLine->isNegative();`

isVertical [NEW]

Parameter: None

**Return value
data type:** Boolean

Remarks: Returns whether the line is vertical or not.

Example: `$svVerticalTmp = $a2wLine->isVertical();`

new [NEW]

Parameter: None

Return value data type: Instance of a line object of type a2w::Line

Remarks: Returns a new allocated line instance.

Example: `$a2wLine = new a2w::Line();`

remove [NEW]

Parameter: None

Return value data type: Void

Remarks: Removes the line from its container, but positioning is still done in the coordinate system.

Example: `$a2wLine->remove();`

setColor [NEW]

Parameter: Color of type integer

Return value data type: Void

Remarks: Sets the color of the line.

Example: `$a2wLine->setColor(0x000000FF);`

setHorizontal [NEW]

Parameter: Horizontal flag of type boolean

Return value Void
data type:

Remarks: Sets the horizontal attribute of the line.

Example: `$a2wLine->setHorizontal (1);`

setLength [NEW]

Parameter: Line length of type integer

Return value Void
data type:

Remarks: Sets the line length.

Example: `$a2wLine->setLength(150);`

setNegative [NEW]

Parameter: Negative flag of type boolean

Return value Void
data type:

Remarks: Sets the negative attribute of the line.

Example: `$a2wLine->setNegative(1);`

setVertical [NEW]

Parameter: Vertical flag of type boolean

Return value Void
data type:

Remarks: Sets the vertical attribute of the line.

Example: `$a2wLine->setVertical (1);`

setWidth [NEW]

Parameter: Line width of type integer

Return value Void
data type:

Remarks: Sets the line's width.

Example: `$a2wLine->setWidth(2);`

setXPos [NEW]

Parameter: X position of type integer

Return value Void
data type:

Remarks: Set the X position of the line.

Example: `$a2wLine->setXPos(100);`

setYPos [NEW]

Parameter: Y position of type integer

Return value Void
data type:

Remarks: Sets the Y position of the line.

Example: `$a2wLine->setYPos(200);`

a2w::MediumMap [NEW]

The a2w::MediumMap object wraps the AFP2web Kernel object mhtMediumMap and exposes the following interfaces to the AFP2web Scripting Facility.,

getDuplexControl [NEW]

Parameter: None

**Return value
data type:** Integer

Remarks: Returns the duplex control setting, like simple, double or triple.

Example: `$svDupCtrlTmp = $a2wMediumMap->getDuplexControl();`

getFormdefName [NEW]

Parameter: None

**Return value
data type:** Integer

Remarks: Returns the name of the Formdef.

Example: `$svFormdefNameTmp = $a2wMediumMap->getFormdefName();`

getHeight [NEW]

Parameter: None

Return value Integer
data type:

Remarks: Returns the height of the medium map.

Example: `$svHeightTmp = $a2wMediumMap->getHeight();`

getName [NEW]

Parameter: None

Return value String
data type:

Remarks: Returns the name of medium map.

Example: `$svNameTmp = $a2wMediumMap->getName();`

getNupControl [NEW]

Parameter: None

Return value Integer
data type:

Remarks: Returns the N-up control setting and the return value must be in between 1 to 4.

Example: `$svNupCtrlTmp = $a2wMediumMap->getNupControl();`

getResolution [NEW]

Parameter: None

**Return value
data type:** Integer

Remarks: Returns the resolution of medium

Example: `$svResolutionTmp = $a2wMediumMap->getResolution();`

getWidth [NEW]

Parameter: None

**Return value
data type:** Integer

Remarks: Returns the width of the medium map.

Example: `$svWidthTmp = $a2wMediumMap->getWidth();`

a2w::NOP (mhtNOP [DEPRECATED])

The a2w::NOP object wraps the AFP2web Kernel object mhtNOP and exposes the following interfaces to the AFP2web Scripting Facility.

getEBCDICValue

Parameter: None

**Return value
data type:** String

Remarks: Gets the NOP's EBCDIC value

Example: `$svEBCDICValueTmp = $a2wNOP->getEBCDICValue();`

getLength

Parameter: None

**Return value
data type:** Integer

Remarks: Gets the NOP's length

Example: `$svLengthTmp = $a2wNOP->getLength();`

getValue

Parameter: None

**Return value
data type:** String

Remarks: Gets the NOP's value.

Example: `$svValueTmp = $a2wNOP->getValue();`

new [NEW]

Parameter: None

**Return value
data type:** NOP instance of type a2w::NOP

Remarks: Returns newly allocated instance of a2w::NOP

Example: `$a2wNOP = new a2w::NOP();`

remove [NEW]

Parameter: None

**Return value
data type:** Void

Remarks: Removes NOP from list of page/document NOPs

Example: `$a2wNOP->remove();`

setValue

Parameter: NOP value of type string

Return value Void
data type:

Remarks: Set nop value

Example: `$a2wNOP->setValue("Created by AFP2web");`

a2w::Overlay (mhtResOverlay [DEPRECATED])

The a2w::Overlay object wraps the AFP2web Kernel object mhtResOverlay and exposes the following interfaces to the AFP2web Scripting Facility.

getFirstLine [NEW]

Parameter: None

Return value data type: Line instance of type a2w::Line

Remarks: Returns the first line object of overlay; The return type is a pointer to "mhtLineObject".

Example:

```
$a2wLineTmp = $a2wOverlay->getFirstLine();  
while ( $a2wLineTmp != 0 ){  
    #---- Access line info  
    ...  
    #---- Get next line  
    $a2wLineTmp = $a2wOverlay->getNextLine();  
}
```

getFirstOverlay [NEW]

Parameter: None

Return value data type: Overlay instance of type a2w::Overlay

Remarks: Returns the first included overlay resource; The return type is a pointer to "mhtResOverlay".

Example:

```

$a2wOverlayTmp = $a2wOverlay->getFirstOverlay();
while $a2wOverlayTmp != 0 ){
#---- Access overlay info
...
#---- Get next overlay
$a2wOverlayTmp = $a2wOverlay->getNextOverlay();
}

```

getFirstPageSegment [NEW]

Parameter: None

Return value data type: Page segment instance of type a2w::PSEG

Remarks: Returns the first included page segment resource; The return type is a pointer to "mhtResPSEG".

Example:

```

$a2wPSEGTmp = $a2wOverlay->getFirstPageSegment();
while $a2wPSEGTmp != 0 ){
#---- Access page segment info
...
#---- Get next page segment
$a2wPSEGTmp = $a2wOverlay->getNextPageSegment();
}

```

getFirstText [MODIFIED]

Parameter: None

Return value data type: Text instance of type a2w::Text

Remarks: Get the first text object of overlay.
Note: This method was previously called "getFirstTextObject".

Example:

```
$a2wTextTmp = $a2wOverlay->getFirstText();  
while ($a2wTextTmp != 0 ){  
#---- Access text info  
...  
#---- Get next text  
$a2wTextTmp = $a2wOverlay->getNextText();  
}
```

getHeight [NEW]

Parameter: None

**Return value
data type:** Integer

Remarks: Returns the height of the overlay.

Example: \$svHeightTmp = \$a2wOverlay->getHeight();

getIncludedXPosition [NEW]

Parameter: None

**Return value
data type:** Integer

Remarks: Returns the X position of this overlay included in the page.

Example: \$svInclXPosTmp = \$a2wOverlay->getIncludedXPosition();

getIncludedYPosition [NEW]

Parameter: None

Return value data type: Integer

Remarks: Returns the Y position of this overlay included in the page.

Example: `$svInclYPosTmp = $a2wOverlay->getIncludedYPosition();`

getName

Parameter: None

Return value data type: String

Remarks: Returns the name of the overlay.

Example: `$svNameTmp = $a2wOverlay->getName();`

getNextLine [NEW]

Parameter: None

Return value data type: Line instance of type a2w::Line

Remarks: Returns the next line object of the overlay; The return type is a pointer to "mhtLineObject".

Example:

```
$a2wLineTmp = $a2wOverlay->getFirstLine();
while $a2wLineTmp != 0 ){
#---- Access line info
...
#---- Get next line
$a2wLineTmp = $a2wOverlay->getNextLine();
}
```

getNextOverlay [NEW]

Parameter: None

Return value data type: Overlay instance of type a2w::Overlay

Remarks: Returns the next included overlay resource (used to traverse the list of included overlays); The return type is a pointer to "mhtResOverlay".

Example:

```
$a2wOverlayTmp = $a2wOverlay->getFirstOverlay();  
while $a2wOverlayTmp != 0 ){  
#---- Access overlay info  
...  
#---- Get next overlay  
$a2wOverlayTmp = $a2wOverlay->getNextOverlay();  
}
```

getNextPageSegment [NEW]

Parameter: None

Return value data type: Page segment instance of type a2w::PSEG

Remarks: Returns the next included page segment resource (used to traverse the list of included overlays); The return type is a pointer to "mhtResPSEG".

Example:

```
$a2wPSEGTmp = $a2wOverlay->getFirstPageSegment();  
while $a2wPSEGTmp != 0 ){  
#---- Access page segment info  
...  
#---- Get next page segment  
$a2wPSEGTmp = $a2wOverlay->getNextPageSegment();  
}
```

getNextText [MODIFIED]

Parameter: None

Return value Text instance of type a2w::Text
data type:

Remarks: Get the next text object of overlay (used to iterate the list of overlay text objects after calling "getFirstText").
Note: This method was previously called "getNextTextObject".

Example:

```
$a2wTextTmp = $a2wOverlay->getFirstText();
while ($a2wTextTmp != 0 ){
#---- Access text info
...
#---- Get next text
$a2wTextTmp = $a2wOverlay->getNextText();
}
```

getResolution [NEW]

Parameter: None

Return value Integer
data type:

Remarks: Returns the resolution of the overlay.

Example: \$svResTmp = \$a2wOverlay->getResolution();

getWidth [NEW]

Parameter: None

Return value Integer
data type:

Remarks: Returns the width of the overlay

Example: \$svWidthTmp = \$a2wOverlay->getWidth();

a2w::PSEG [NEW]

The a2w::PSEG object wraps the AFP2web Kernel object mhtResPSEG and exposes the following interfaces to the AFP2web Scripting Facility.

getIncludedXPosition [NEW]

Parameter: None

**Return value
data type:** Integer

Remarks: Returns included X position of this page segment in page

Example: `$svInclXPosTmp = $a2wPSEG->getIncludedXPosition();`

getIncludedYPosition [NEW]

Parameter: None

**Return value
data type:** Integer

Remarks: Returns included Y position of this page segment in page

Example: `$svInclYPosTmp = $a2wPSEG->getIncludedYPosition();`

getName [NEW]

Parameter: None

Return value String
data type:

Remarks: Returns page segment name

Example: \$svNameTmp = \$a2wPSEG->getName();

a2w::Page (mhtPtocaPage [DEPRECATED], mhtLPDPage [DEPRECATED])

The a2w::Page object wraps the AFP2web Kernel objects mhtPtocaPage, mhtTIFFPage, mhtLPDPage and exposes following interfaces to the AFP2web Scripting Facility.

addAnnotation

Parameter: X position of type integer
 Y position of type integer
 Width of type integer
 Height of type integer
 URL text of type string
 Border width of type integer
 Color of type integer
 Flags of type integer

Returnvalue Void
data type:

Remarks: Add annotation to page, useful when PDF output is generated else ignored

Example: `$a2wPage->addAnnotation(0, 0, 100, 10, "MAAS", 2, 0x00FF0000, 0);`

addBookmark [NEW]

Parameter: Bookmark name of type string
 Bookmark value of type string

**Return value
data type:** Void

Remarks: Add bookmark to page

Example: `$a2wPage->addBookmark("Software", "AFP2web");`

addImage [NEW]

Parameter: Image name of type string
 X include position of type integer
 Y include position of type integer
 Presentation Width of type integer
 Presentation Height of type integer
 Rotation of type integer

**Return value
data type:** Void

Remarks: Add raster image to page

Example: `$a2wPage->addImage("BGFORM", 0, 100, 1000, 1500, 0);`

addIndex [DEPRECATED]

Parameter: Index name of type string
 Index value of type string

**Return value
data type:** Void

Remarks: Add index to page, deprecated by new "addIndex" interface explained later in this chapter 14.35

Example: `$a2wPage->addIndex("Producer", "Maas");`

addIndex [NEW]

Parameter: Index instance of type *a2w::Index*

Return value Void
data type:

Remarks: Add index to page

Example: `#---- Create new index
$a2wIndexTmp = new a2w::Index();
#---- Set index info
$a2wIndexTmp->setName("Producer");
$a2wIndexTmp->setValue("Maas");
#---- Add index to page
$a2wPage->addIndex($a2wIndexTmp);`

addLine [NEW]

Parameter: Line instance of type *a2w::Line*

Return value Void
data type:

Remarks: Add line to page

Example: `#---- Create new line
$a2wLineTmp = new a2w::Line();
#---- Set line info
$a2wLineTmp->setLength(100);
...
#---- Add line to page
$a2wPage->addLine($a2wLineTmp);`

addNOP [NEW]

Parameter: NOP instance of type `a2w::NOP`
 Before "BPG" flag of type Boolean (True means before BPG, default is false)

Return value Void
data type:

Remarks: Add NOP to page

Example:

```
#---- Create new NOP
$a2wNOPTmp = new a2w::NOP();
#---- Set NOP value
$a2wNOPTmp->setValue( "Created by AFP2web" );
#---- Add NOP to page
$a2wPage->addNOP( $a2wNOPTmp );
```

addObject [DEPRECATED]

Parameter: Graphic object instance of type `a2w::Text`

Return value Void
data type:

Remarks: Adds a text object to page, deprecated by the new interface "addText"

Example: `$a2wPage->addObject($a2wTextTmp);`

addOverlay [NEW]

Parameter: Overlay name of type string
 X include position of overlay of type integer
 Y include position of overlay of type integer

Return value Void
data type:

Remarks: Add overlay to page

Example: \$a2wPage->addOverlay("01MEDF01", 0, 100);

addPSEG [NEW]

Parameter: Page segment name of type string
 X include position of type integer
 Y include position of type integer
 Presentation Width of type integer
 Presentation Height of type integer
 Rotation of type integer

Return value Void
data type:

Remarks: Add page segment to page

Example: \$a2wPage->addPSEG("S1MAAS", 0, 100, 1000, 1500, 0);

addText [NEW]

Parameter: Text instance of type *a2w::Text*

Return value Void
data type:

Remarks: Add text to page

Example:

```
#---- Create new text
$a2wTextTmp = new a2w::Text();
#---- Set text info
$a2wTextTmp ->setText( "Maas Test" );
...
#---- Add text to page
$a2wPage->addText( $a2wTextTmp );
```

getAppliedMediumMap [NEW]

Parameter: None

**Return value
data type:** String

Remarks: Returns medium map applied on given page. Return type is pointer to "mhtMediumMap"

Example: `$a2wMediumMap = $a2wPage->getAppliedMediumMap();`

getFirstIndex

Parameter: None

**Return value
data type:** Index instance of type `a2w::Index`

Remarks: Get first index of page

Example:

```
$a2wIndexTmp = $a2wPage->getFirstIndex();
while ( $a2wIndexTmp != 0 ){
#---- Access index info
...
#---- Get next index
$a2wIndexTmp = $a2wPage->getNextIndex();
}
```

getFirstLine [NEW]

Parameter: None

**Return value
data type:** Line instance of type a2w::Line

Remarks: Returns first line object of page. Return type is pointer to "mhtLineObject"

Example:

```
$a2wLineTmp = $a2wPage->getFirstLine();
while ( $a2wLineTmp != 0 ){
#---- Access line info
...
#---- Get next line
$a2wLineTmp = $a2wPage->getNextLine();
}
```

getFirstMediumOverlay [NEW]

Parameter: None

**Return value
data type:** Overlay instance of type a2w::Overlay

Remarks: Returns first medium applied overlay. Return type is pointer to "mhtResOverlay"

Example:

```
$a2wOverlayTmp = $a2wPage->getFirstMediumOverlay();
while ( $a2wOverlayTmp != 0 ){
#---- Access overlay info
...
#---- Get next overlay
$a2wOverlayTmp = $a2wPage->getNextMediumOverlay();
}
```

getFirstNOP

Parameter: None

Return value NOP instance of type a2w::NOP
data type:

Remarks: Get first nop of page

Example:

```

$a2wNOPTmp = $a2wPage->getFirstNOP();
while ( $a2wNOPTmp != 0 ){
#---- Access NOP info
...
#---- Get next NOP
$a2wNOPTmp = $a2wPage->getNextNOP();
}

```

getFirstOverlay

Parameter: None

Return value Overlay instance of type a2w::Overlay
data type:

Remarks: Get first page included overlay

Example:

```

$a2wOlyTmp = $a2wPage->getFirstOverlay();
while ( $a2wOlyTmp != 0 ){
#---- Access overlay info
...
#---- Get next overlay
$a2wOlyTmp = $a2wPage->getNextOverlay();
}

```

getFirstPageSegment [NEW]

Parameter: None

Return value Page segment instance of type a2w::PSEG
data type:

Remarks: Returns first page included page segment. Return type is pointer to "mhtResPSEG"

a2w::Page (*mhtmlPage* [DEPRECATED], *mhtmlPage* [DEPRECATED])

Example:

```
$a2wPSEGTmp = $a2wPage->getFirstPageSegment();
while ( $a2wPSEGTmp != 0 ){
#---- Access page segment info
...
#---- Get next page segment
$a2wPSEGTmp = $a2wPage->getNextPageSegment();
}
```

getFirstText [MODIFIED]

Parameter: None

Return value Text instance of type *a2w::Text*
data type:

Remarks: Get first text object of page, previously called as "getFirstTextObject"

Example:

```
$a2wTextTmp = $a2wPage->getFirstText();
while ( $a2wTextTmp != 0 ){
#---- Access Text info
...
#---- Get next text
$a2wTextTmp = $a2wPage->getNextText();
}
```

getHeight

Parameter: None

Return value Integer
data type:

Remarks: Get page height

Example: `$svHeightTmp = $a2wPage->getHeight();`

getId

Parameter: None

**Return value
data type:** Integer

Remarks: Get page id

Example: `$svIdTmp = $a2wPage->getId();`

getName [NEW]

Parameter: None

**Return value
data type:** String

Remarks: Returns page name

Example: `$svNameTmp = $a2wPage->getName();`

getNextIndex

Parameter: None

**Return value
data type:** Index instance of type `a2w::Index`

Remarks: Get next index of page (used to iterate list of page indexes after calling "getFirstIndex")

a2w::Page (*mhtPlocaPage* [DEPRECATED], *mhtLPDPPage* [DEPRECATED])

Example:

```
$a2wIndexTmp = $a2wPage->getFirstIndex();
while ( $a2wIndexTmp != 0 ){
#---- Access index info
...
#---- Get next index
$a2wIndexTmp = $a2wPage->getNextIndex();
}
```

getNextLine [NEW]

Parameter: None

Return value Line instance of type *a2w::Line*
data type:

Remarks: Returns next line object of page (used to traverse the list of page lines). Return type is pointer to “mhtLineObject”

Example:

```
$a2wLineTmp = $a2wPage->getFirstLine();
while ( $a2wLineTmp != 0 ){
#---- Access line info
...
#---- Get next line
$a2wLineTmp = $a2wPage->getNextLine();
}
```

getNextMediumOverlay [NEW]

Parameter: None

Return value Overlay instance of type *a2w::Overlay*
data type:

Remarks: Returns next medium applied overlay (used to traverse the list of medium applied overlays). Return type is pointer to “mhtResOverlay”

Example:

```

$a2wOverlayTmp = $a2wPage->getFirstMediumOverlay();
while ( $a2wOverlayTmp != 0 ){
#---- Access overlay info
...
#---- Get next overlay
$a2wOverlayTmp = $a2wPage->getNextMediumOverlay();
}

```

getNextNOP

Parameter: None

Return value data type: NOP instance of type a2w::NOP

Remarks: Get next NOP of page (used to iterate list of page nops after calling "getFirstNOP")

Example:

```

$a2wNOPTmp = $a2wPage->getFirstNOP();
while ( $a2wNOPTmp != 0 ){
#---- Access NOP info
...
#---- Get next NOP
$a2wNOPTmp = $a2wPage->getNextNOP();
}

```

getNextOverlay

Parameter: None

Return value data type: Overlay instance of type a2w::Overlay

Remarks: Get next page included overlay (used to iterate list of page indexes after calling "getFirstOverlay")

a2w::Page (*mhtPlocaPage* [DEPRECATED], *mhtLPDPPage* [DEPRECATED])

Example:

```
$a2wOverlayTmp = $a2wPage->getFirstOverlay();
while ( $a2wOverlayTmp != 0 ){
#---- Access overlay info
...
#---- Get next overlay
$a2wOverlayTmp = $a2wPage->getNextOverlay();
}
```

getNextPageSegment [NEW]

Parameter: None

Return value Page segment instance of type *a2w::PSEG*
data type:

Remarks: Returns next page included page segment (used to traverse the list of page included page segments). Return type is pointer to “mhtResPSEG”

Example:

```
$a2wPSEGTmp = $a2wPage->getFirstPageSegment();
while ( $a2wPSEGTmp != 0 ){
#---- Access page segment info
...
#---- Get next page segment
$a2wPSEGTmp = $a2wPage->getNextPageSegment();
}
```

getNextText [MODIFIED]

Parameter: None

Return value Text instance of type *a2w::Text*
data type:

Remarks: Get next text object of page (used to iterate list of page text object after calling “getFirstText”), previously called as “getNextTextObject”

Example:

```

$a2wTextTmp = $a2wPage->getFirstText();
while ( $a2wTextTmp != 0 ){
#---- Access Text info
...
#---- Get next text
$a2wTextTmp = $a2wPage->getNextText();
}

```

getOutputBuffer.[NEW]

Parameter: None

**Return value
data type:** Buffer of type byte*

Remarks: Gets the page output buffer.

Example: \$svPageBufferTmp = \$a2wPage->getOutputBuffer();

getOutputBufferLength.[NEW]

Parameter: None

**Return value
data type:** Integer

Remarks: Gets the length of the page output buffer.

Example: \$svPageBufferLenTmp = \$a2wPage->getOutputBufferLength();

getOutputFilename.[NEW]

Parameter: None

**Return value
data type:** String

Remarks: Gets the name of the output file of this page.

Example: `$svOutputFilenameTmp = $a2wPage->getOutputFilename();`

getPageGroupName [NEW]

Parameter: None

**Return value
data type:** String

Remarks: Returns name of page group, which contain this page

Example: `$svPageGroupNameTmp = $a2wPage->getPageGroupName();`

getParseId

Parameter: None

**Return value
data type:** Integer

Remarks: Get page parse id (sequence id of page as in input spool)

Example: `$svParseIdTmp = $a2wPage->getParseId();`

getResolution

Parameter: None

**Return value
data type:** Integer

Remarks: Get resolution

Example: `$svResTmp = $a2wPage->getResolution();`

getSimpleFilename.[NEW]

Parameter: None

**Return value
data type:** String

Remarks: Gets the simple file name of the output file of this page.

Example: `$svSimpleFilenameTmp = $a2wPage->getSimpleFilename();`

getSize.[NEW]

Parameter: None

**Return value
data type:** Long

Remarks: Gets the size of the output file of this page.

Example: `$svOutputSizeTmp = $a2wPage->getSize();`

getType

Parameter: None

Return value Integer
data type:

Remarks: Get page type (AFP or LPD)

Example: `$svTypeTmp = $a2wPage->getType();`

getWidth

Parameter: None

Return value Integer
data type:

Remarks: Get page width

Example: `$svWidthTmp = $a2wPage->getWidth();`

includeObject [DEPRECATED]

Parameter: Object type of type byte
Object name of type string
X position of type integer
Y position of type integer
Rotation of type integer
Object sub type of type string

Return value Void
data type:

Remarks: Include an object (an overlay, image etc). Useful to apply some background forms to the page, deprecated by new interfaces like "addOverlay", "addPageSegment" and "addImage"

Example: `$a2wPage->includeObject(0x92, "BGFORM", 0, 0, 0, "JPEG");`

isConstantBack

Parameter: None

Return value data type: Boolean

Remarks: Returns true if page is medium applied constant back

Example: `$svConstantBackTmp = $a2wPage->isConstantBack();`

isOverlayFound

Parameter: Overlay name of type string

Return value data type: Boolean

Remarks: Check whether given overlay is included in page or not

Example: `$svOverlayFoundTmp = $a2wPage->isOverlayFound("01MEDF01");`

new [NEW]

Parameter: Page type of type integer and value is interpreted as given in the table:

Page Type	Value	Description
-----------	-------	-------------

1	AFP Page
2	LPD Page
3	TI FF Page

Return value

Page instance of type a2w::Page

data type:

Remarks:

Creates new page instance and returns the same

Example:

\$a2wPageTmp = new a2w::Page(1);

setBackgroundColor

Parameter:

Page color of type integer

Return value

Void

data type:

Remarks:

Set page background color

Example:

\$a2wPage->setBackgroundCol or(0x00FF0000);

setHeight

Parameter:

Page height of type integer

Return value

Void

data type:

Remarks:

Set page height

Example:

\$a2wPage->setHei ght(584);

setOutputFileName.[NEW]

Parameter: Output filename of type string

**Return value
data type:** Void

Remarks: Sets the name of the output file for this page.

Example: `$a2wPage->setOutputFileName ("MaasSample.tif");`

setResolution

Parameter: Page resolution of type integer

**Return value
data type:** Void

Remarks: Set page resolution

Example: `$a2wPage->setResolution(240);`

setWidth

Parameter: Page width of type integer

**Return value
data type:** Void

Remarks: Set page width

Example: `$a2wPage->setWidth(840);`

a2w::Text (mhtTextObject [DEPRECATED])

“a2w::Text” module wraps “mhtTextObject” kernel object and expose following interfaces to scripting facility

getAngle

Parameter:	None
-------------------	------

Return value data type:	Integer
------------------------------------	---------

Remarks:	Get angle (rotation)
-----------------	----------------------

Example:	<code>\$svAngleTmp = \$a2wText->getAngle();</code>
-----------------	---

getColor [NEW]

Parameter:	None
-------------------	------

Return value data type:	Integer
------------------------------------	---------

Remarks:	Get text color of 4 bytes length (as given in below format)
-----------------	---



Example: `$svColorTmp = $a2wText->getColor();`

getEBCDICText

Parameter: None

**Return value
data type:** String

Remarks: Get EBCDIC value of text object

Example: `$svEBCDICTextTmp = $a2wText->getEBCDICText();`

getFont [NEW]

Parameter: None

**Return value
data type:** Font instance of type a2w::Font

Remarks: Returns font used to render this text, return type is pointer to "mhtAFPFont"

Example: `$a2wFontTmp = $a2wText->getFont();`

getMappedFontLocalId [NEW]

Parameter: None

**Return value
data type:** Integer

Remarks: Get text-applied font's local map id

Example: `$svFontLocalIdTmp = $a2wText->getMappedFontLocalId();`

getText

Parameter: None

**Return value
data type:** String

Remarks: Get value of text object

Example: `$svTextTmp = $a2wText->getText();`

getTextLen

Parameter: None

**Return value
data type:** Integer

Remarks: Get value length of text object

Example: `$svTextLenTmp = $a2wText->getTextLen();`

getXPos

Parameter: None

**Return value
data type:** Integer

Remarks: Get X position of text object

Example: `$svXPosTmp = $a2wText->getXPos();`

getYPos

Parameter: None

**Return value
data type:** Integer

Remarks: Get Y position of text object

Example: `$svYPosTmp = $a2wText->getYPos();`

new

Parameter: None

**Return value
data type:** Text instance of type `a2w::Text`

Remarks: Allocate new text instance

Example: `$a2wText = new a2w::Text();`

remove [NEW]

Parameter: None

**Return value
data type:** Void

Remarks: Removes text from presentation, but positioning text object on output coordinate system is done still

Example: `$a2wText->remove();`

setAngle

Parameter: Angle of type integer

**Return value
data type:** Void

Remarks: Set angle (rotation) of text object

Example: `$a2wText->setAngle(90);`

setColor

Parameter: Color of type integer

**Return value
data type:** Void

Remarks: Set color of text object

Example: `$a2wText->setColor(0x00FF0000);`

setFont

Parameter: Font instance of type a2w::Font

**Return value
data type:** Void

Remarks: Set font of text object

Example:

```
#---- Create font
$a2wFontTmp = new a2w::Font();
#---- Set font details
$a2wFont->setName( "Verdana" );
...
#---- Set above created font as font in text
$a2wText->setFont( $a2wFontTmp );
```

setText

Parameter: Text of type string

**Return value
data type:** Void

Remarks: Set value of text object

Example: `$a2wText->setText("Maas Test");`

setTextLen

Parameter: Text length of type integer

**Return value
data type:** Void

Remarks: Set value length of text object

Example: `$a2wText->setTextLen(9);`

setXPos

Parameter: X position of type integer

Return value Void
data type:

Remarks: Set X position of text object

Example: `$a2wText->setXPos(100);`

setYPos

Parameter: Y position of type integer

Return value Void
data type:

Remarks: Set Y position of text object

Example: `$a2wText->setYPos(200);`

5.10 AFP2web Meldungen

Allgemeine Hinweise zu Meldungen, Rückgabewerte, und Error-Dateien

afp2web.exe liefert einen Rückgabewert nach Beendigung der Verarbeitung.

AFP2web schreibt auch Fehlermeldungen in eine separate Textdatei. Sie finden diese Datei in das LOG-Verzeichnis. Die Dateinamen beginnt mit der Zeichenfolge "error_", ergänzt um den Zeitstempel. Beispiel: error_20000413_101127.txt

AFP2web übergibt einen Rückgabewert zur Fehlermeldung:

- Der Rückgabewert 0 ist Hinweis, dass ein Fehlerfall nicht vorliegt. Das Präfix "I" in der ID bedeutet, die Meldung ist informativ.
- Ein negativer Rückgabewert weist auf einen Verarbeitungsfehler hin. Der Wert korrespondiert mit dem numerischen Teil der Meldungs-ID. Beispiel: Ein Rückgabewert -23 weist auf den Fehler E023 hin. Das Präfix "E" in der ID weist auf einen Fehler hin, "W" auf eine Warnmeldung.

Hinweis: Wenn Sie eine Fehlermeldung nicht verstehen, kontaktieren Sie bitte die Maas High Tech Software GmbH.

AFP2web Fehlermeldungen

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
Info-Meldung	
0	I000: Process completed Prozess ist erfolgreich beendet.
	I001: End of spool:<SpoolFileName> Die Spooldatei ist erfolgreich abgearbeitet.
0	I022: User requested no. of pages parsed. Ignoring other pages Meldung tritt auf, wenn die Aufrufoption -pp verwendet wird. Keine Aktion erforderlich.
0	I083: User requested no. of documents parsed. Ignoring other documents Meldung tritt auf, wenn die Aufrufoption -ed verwendet wird. Keine Aktion erforderlich.
Warnungen (Meldungen, die den Prozess nicht unterbrechen))	
	W016: Missing mandatory value : <Value>. MO:DCA-spezifischer Warnung. AFP Dokument prüfen. Wenn Sie das Problem nicht beheben können, kontaktieren Sie bitte Maas High Tech Software GmbH.
	W060: Invalid SFI. <StreamLocationAndName> Offset=<HexOffset> MO:DCA-spezifischer Warnung. AFP Dokument prüfen
	W093: Insufficient Font Pattern data for GCGID <GCGIDName> for resource <ResName>. Available Bytes: <DataLength> Required Bytes: <Req.DataLength>. FOCCA-spezifischer Warnung. Prüfen Sie bitte das Dokument mit der angegebenen Font-Ressource.
	W154: Invalid Index Path (<InfoText>) Der Pfad für die Ausgabe von Indexdateien ist ungültig. Stellen Sie sicher, dass der Pfad zu der Indexdatei vorhanden ist.

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
Fehlermeldungen	
-1	E001: Unknown error. Unbekannt Bitte kontaktieren Sie Maas High Tech Software GmbH.
-2	E002: Unknown exception occurred, leaving... Programmfehler Bitte kontaktieren Sie Maas High Tech Software GmbH.
-3	E003: AFP2web is not licensed for <OS name Input format Output format Scripting Facility>. Please contact afp2web@maas.de Lizenz ist nicht gültig für eine der folgenden Optionen: Betriebssystem, Eingabeformat, Ausgabeformat, Scripting Facility. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-4	E004: <FileType> File <FileName> not Found. AFP2web kann die Ressource oder Indexdatei nicht finden. Stellen Sie sicher, dass die externe Ressource oder Indexdatei in dem vorgegebenen Pfad vorhanden ist.
-5	E005: Missing End Page Group SFI. File Name: <FileName> MO:DCA-spezifischer Fehler AFP Dokument prüfen
-6	E006: Missing End Document SFI. File Name: <FileName> MO:DCA-spezifischer Fehler AFP Dokument prüfen
-7	E007: Illegal reserved value. (at <HexOffset>) MO:DCA-spezifischer Fehler AFP Dokument prüfen
-8	E008: AFP font representation object is null. FOCA-spezifischer Fehler Prüfen Sie bitte die AFP Font-Ressource
-9	E009: Missing Mandatory Triplet. (at <HexOffset>) MO:DCA-spezifischer Fehler AFP Dokument prüfen
-10	E010: Valid spool begining not found. File Name: <FileName> MO:DCA-spezifischer Fehler AFP Dokument prüfen

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-11	E011: Character metrics missing for <Degree> degree rotation. Charset-Name: <CharsetName> FOCCA-spezifischer Fehler. Prüfen Sie bitte das FOCA AFP Dokument.
-12	E012: AFP2web is not built for<FormatType> <"Input"/"Output" Format>. Please contact Maas High Tech Software GmbH at afp2web@maas.de Die Kombination von Schlüsselwerten in der INI-Datei ist nicht gültig. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-13	E013: <MHT Image Library Error> Fehler aus der MHT Image Bibliothek. Prüfen Sie die Fehlermeldungen aus der MHT Image Bibliothek.
-14	E014: Missing End Page SFI. File Name: <FileName> MO:DCA-spezifischer Fehler AFP Dokument prüfen
-15	E015: Missing End Overlay SFI. File Name: <FileName> MO:DCA-spezifischer Fehler AFP Dokument prüfen
-17	E017: Message id <MsgID> define into ErrorCode enum structure, but Error message missing within achErrorMsg[]. Check code. Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-18	E018: <CompressionName> compressed <ImageType> images is not supported. AFP2web unterstützt nicht die angegebene Kompression für den Grafiktyp. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-19	< Bits/Pixel> Bits/Pixel <ImageType> image is not supported AFP2web unterstützt nicht die angegebene Dateityp mit den angegebenen Bits/Pixel. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-20	E020: Unable to find any substitutable external font for [CF=<CodedFont name>][CS=<Charset name>][TF=<Typeface name>] AFP2web findet keine externe Schrift, die als Ersatzschrift verwendet werden kann. Überprüfen Sie, ob eine Font-Datei vorhanden ist, mit der angegebenen Benennung für Typeface übereinstimmt. Der Ordner für die externe Font-Datei wird festgelegt mit dem INI Parameter "extfont" bzw. mit der Kommandozeilenoption "-fp"

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-21	E021: Java Exception occurred while reading stream (<Stream Name>). Msg:<InfoText> AFP2web kann den Java Eingabe Datenstrom nicht lesen. Der Grund ist in dem Meldetext angegeben. Überprüfen Sie ob es möglich, ist von der Quelle des Eingabe Datenstroms zu lesen.
-23	E023: Input file (<InfoText>) not found. Eingabe-Datei kann nicht gefunden werden. Verzeichnis, Aufrufparameter überprüfen
-24	E024: Invalid Output Format: <Output format> AFP2web akzeptiert nur ein gültiges Ausgabeformat. Überprüfen Sie Ihre Auswahl für das Ausgabeformat.
-25	E025: Insufficient Memory - Unable to create display Device context. Möglicherweise zu hohe Auflösung für TIFF-Konvertierung Bitte erweitern Sie ihren RAM-Speicher.
-26	E026: Insufficient Memory - Unable to create Memory Device context. Möglicherweise zu hohe Auflösung für TIFF-Konvertierung Bitte erweitern Sie ihren RAM-Speicher.
-27	E027: Insufficient memory while allocating Space for Bitmap data. Möglicherweise zu hohe Auflösung für TIFF-Konvertierung Bitte erweitern Sie ihren RAM-Speicher.
-28	E028: Insufficient memory - Unable to create bitmap for a page (Error-Code: <Last Error Code of Win32 API>) Möglicherweise zu hohe Auflösung für die Konvertierung nach TIFF/JPG/PNG. Bitte erweitern Sie ihren RAM-Speicher oder reduzieren Sie die Auflösung (mit der Aufrufoption -pr).
-29	E029: Could not install outline font to system. Character set: <Charset Name> CodePage:<CP Name> Reason:<Reason> AFP2web kann die von dem AFP2web Outline Font erzeugte Type 1 Schrift im System nicht installieren. Stellen Sie sicher, dass der Benutzer über die Berechtigung verfügt, externe Schriften im System zu installieren.
-30	E030: Unable to read a TIFF header of <Filename> (Bad TIFF). Diese Meldung tritt auf, beim Versuch ein TIFF Dokument nach PDF zu konvertieren. Das TIFF Dokument kann nicht in diesem Format gelesen werden. Bitte überprüfen Sie das TIFF Dokument. Wenn Sie das Problem nicht beheben können, konsultieren Sie bitte Maas High Tech Software GmbH.

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-31	E031: Could not install external font to system. Font File: <FileName>, Reason: <Reason> AFP2web kann die Type 1 / TrueType Schrift nicht im System installieren. Stellen Sie sicher, dass der Benutzer über die Berechtigung verfügt, externe Schriften im System zu installieren.
-32	E032: Input file (<InfoText>) is not a TIFF file. Die Eingabedatei muß die Dateierweiterung „.tif“ oder „.tiff“ haben. Verwenden Sie die erforderliche Dateierweiterung für TIFF Dateien.
-33	E033: Unhandled/Unknown compression for image (<ImageName>, <ImageType>) AFP2web unterstützt nicht die angegebene Komprimierung. Kontaktieren Sie bitte Maas High Tech Software GmbH.
-34	E034: Color input image <<ImageType>, <ImageName>, <Compression>> is not supported for <OutputFormat> output AFP2web unterstützt nicht als Eingabe die farbigen Grafiken mit der angegebenen Kompression für das gewählte Ausgabeformat. Kontaktieren Sie bitte Maas High Tech Software GmbH.
-35	E035: Java Input Stream is null for (<StreamName>) Ein Eingabestrom mit NULL als Inhalt wurde an AFP2web übergeben. Stellen Sie sicher, dass ein gültiger Eingabestrom an die Java SDK übergeben wird.
-36	E036: Java <MethodName> Method Handle is null for (<StreamName>) AFP2web konnte einen Handle für die angegebene Methode von der Java Laufzeitumgebung nicht bekommen. Überprüfen Sie bitte die Version der installierten Java-Umgebung. Erforderlich ist zumindest die Version, die als Systemvoraussetzung im AFP2web Benutzerhandbuch aufgeführt ist.
-37	E037: Input stream does not support random access. Setting offset failed for stream(<StreamName>). Der Datenstrom in der Eingabe enthält ungültiges AFP. AFP2web sucht nach gültigen AFP Daten und setzt den Wert für das Offset direkt im Datenstrom. Diese Fehlermeldung wird ausgegeben, wenn der Datenstrom es nicht erlaubt, den Offset-Wert direkt zu setzen (Stichwort: Random Access). Überprüfen Sie bitte den Eingabe-Datenstrom und stellen Sie sicher, dass dieser den freien Zugriff (Random Access) unterstützt.
-38	E038: Unable to create bitmap for rasterization of AFP font <Charset Name>. (ErrorCode: <Errorcode>, Msg: <ErrorMessage>) AFP2web ist nicht in der Lage, eine Bitmap Repräsentation der AFP Schrift zu erzeugen. Überprüfen Sie die Fehlermeldung aus AFP2web und treffen Sie die entsprechenden Maßnahmen. Wenn Sie das Problem nicht lösen können, kontaktieren Sie Maas High Tech Software GmbH.

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-39	E039: Conversion of <FontType> font to <FontType> font not yet supported AFP2web unterstützt nicht die Konvertierung der angegebenen Schrifttypen. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-40	E040: Conversion of <FontType> font to <FontType> font failed. Font-Name: <FontName> Reason: <Reason> Die Schrift in der Eingabestrom könnte ungültig sein. Wenn Sie das Problem nicht beheben können, kontaktieren bitte Sie Maas High Tech Software GmbH.
-41	E041: Unique System Id generation failed. (ErrorCode: <ErrorCode>, Msg:<Reason>) AFP2web kann nicht die Informationen ermitteln, die für die Generierung einer eindeutigen Systemidentifikation erforderlich ist. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-42	E042: Invalid Operating System specified on Licensee("<Licensee>") in <PathToInifile>\<IniFileName>. Please contact Maas High Tech Software GmbH at afp2web@maas.de. Die Angaben zur Lizenz sind laut AFP2web INI-Datei nicht gültig. Stellen Sie sicher, dass der Lizenz Text korrekt in der AFP2web INI-Datei eingegeben wurde.
-43	E043: Invalid A2W Edition Id: <EditionID>. Please contact Maas High Tech Software GmbH at afp2web@maas.de. Die Angaben zur Lizenz sind laut AFP2web INI-Datei nicht gültig. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-52	E052: Error reading Repeated Triplets (at <HexOffset>) MO:DCA-spezifischer Fehler AFP Dokument prüfen
-53	E053: NULL Mapcodedfont object in MCFTriplets() Speicherproblem. Kontaktieren Sie bitte Maas High Tech Software GmbH.
-54	E054: Illegal Resource Type in MCF Resource Local Identifier Triplet (at <HexOffset>) MO:DCA-spezifischer Fehler AFP Dokument prüfen
-55	E055: Illegal Local id value in MCF Resource Local Identifier Triplet (at <HexOffset>) MO:DCA-spezifischer Fehler AFP Dokument prüfen

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-56	E056: Error Illegal Page Overlay Type - <OverlayTypeld> (at <HexOffset>) MO:DCA-spezifischer Fehler AFP Dokument prüfen
-57	E057: Not Able to Get PDF Doc Handle. Exiting... Speicherproblem. Erweitern Sie die Speichergrenze, indem Sie den Parameter DocLimit erhöhen.
-58	E058: Resource (<ResName>) not found Pfad zu Ressourcen nicht korrekt. Überprüfen sie die verwendete INI-Datei und die Parameter beim Programmaufruf und stellen Sie sicher, dass der korrekten Pfad für Ressourcen verwendet wird.
-59	E059: Null Data Buffer System-interner Fehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-61	E061: Input Stream Open Failed System-interner Fehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-62	E062: SFI missing. <StreamLocationAndName> Offset=<HexOffset> MO:DCA-spezifischer Fehler AFP Dokument prüfen
-63	E063: SFI data is null System-interner Fehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-65	E065: [CHARSET] Section: Missing or Invalid Font Height (<InfoText>) in "mapping.def" Einstellungen der mapping.def sind fehlerhaft. Die mapping.def prüfen.
-67	E067: Error Invalid Interchange Set Triplet Id <TripletId> (at <HexOffset>) MO:DCA-spezifischer Fehler AFP Dokument prüfen
-68	E068: File read error Fehler beim Versuch, eine der folgenden Dateien zu lesen: INI Datei, mapping.def Datei, ASCII codepage Datei (*.cp). Stellen Sie sicher, dass die erforderlichen Dateien vorhanden sind.
-69	E069: Error while writing <FileName> file Fehler beim Versuch, in die Datei zu schreiben. Stellen Sie sicher, dass der Pfad zu der Datei vorhanden ist und dass AFP2web Schreibzugriffberechtigung erhält.

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-70	E070: Unable to move a file pointer to a specified location. File Name: <FileName> Dateizugriffsfehler Bitte kontaktieren Sie Maas High Tech Software GmbH.
-72	E072: File (<InfoText>) is empty. Fehler beim Lesen der Eingabe-Datei. Die Datei ist leer. Überprüfen Sie bitte die einzulesende Datei.
-73	E073: Unable to read data from the stream: File <FileName>, ErrorMessage: <InfoText> AFP2web kann die angegebene Datei nicht lesen. Stellen Sie sicher, dass AFP2web Lesezugriffsberechtigung erhält.
-74	E074: SFI does not begin with 5A. <StreamLocationAndName> Offset=<HexOffset> Wir unterscheiden zwei Kategorien von AFP Dateien. (1) Mit dem Wert 0x5A Hex vor jedem einzelnen Structured Field Identifier (SFI). (2) Ohne diesem Hex-Wert. Diese Fehlermeldung zeigt an, dass der Hex Wert 0x5a vor dem ersten SFI, nicht aber in allen Fällen erscheint. Bitte überprüfen Sie die AFP Daten.
-75	E075: Unable to write data into the stream: File <FileName>, ErrorMessage: <InfoText> AFP2web kann die angegebene Datei nicht schreiben. Stellen Sie sicher, dass AFP2web Schreibzugriffsberechtigung erhält.
-76	E076: Not able to find a valid SFI within the 4 Kb of data. <StreamLocationAndName> Offset=<HexOffset> AFP2web findet kein gültiges Structured Field Introducer (SFI) innerhalb der 4 Kb Daten nach dem Offset <HexOffset>. Der Vorgang wurde abgebrochen. Überprüfen Sie das AFP Dokument. Wenn Sie das Problem nicht beheben können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-77	E077: Missing/Invalid Licensee(<Licensee>) or Serial Number(<SerialNumber>) in <PathToInifile>\<IniFileName>. Please contact afp2web@maas.de. Die Lizenzinformationen in der INI Datei sind nicht gültig. Bitte geben gültige Werte für Licensee und SerialNr in der INI Datei. Wenn Sie das Problem nicht beheben können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-78	E078: Unable to load a font (<InfoText>) Eine erforderliche Schrift kann nicht geladen werden. Fehler in der Konvertierung nach TIFF (Unix). Bitte überprüfen Sie, ob die gesuchte Schrift installiert wurde.

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-79	E079: Unable to create a pixmap. Fehler in der Konvertierung nach TIFF (Unix). Bitte kontaktieren Sie Maas High Tech Software GmbH.
-80	E080: Image stretching error Fehler in der Konvertierung nach TIFF (Unix). Bitte kontaktieren Sie Maas High Tech Software GmbH.
-81	E081: Out of memory error. Unable to write intermediate file <FileName> Fehler in der Konvertierung nach TIFF (Unix). Bitte kontaktieren Sie Maas High Tech Software GmbH.
-82	E082: Error while writing intermediate file <FileName> Fehler in der Konvertierung nach TIFF (Unix). Bitte kontaktieren Sie Maas High Tech Software GmbH.
-84	E084: Required Resource (<ResName>) of Type (<ResType>) not Found System-interner Fehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-85	E085: X Window Error (<info Text>) Entweder die X Window API schlug fehl oder AFP2web kann nicht auf X Window zugreifen. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-86	E086: PDF Library Error: <InfoText> Fehler aus der Maas PDF Bibliothek. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-87	E087: TIFF Library Error <InfoText> Fehler aus der LIBTIFF Bibliothek. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-88	E088: Scripting Facility Error (rc=<errorcode>): <infotext> AFP2web Scripting Facility kann das Script nicht ausführen. Überprüfen Sie die Kodierung in Ihrem Skript. Wenn Sie das Problem nicht beheben können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-90	E090: Image Library Error: <InfoText> Beim Einlesen der Grafik gibt es Probleme während der Dekomprimierung von IOCA and IOB Images. Bitte kontaktieren Sie Maas High Tech Software GmbH und stellen Sie die Error- und Log-Dateien bereit.
-91	E091: Output buffer of IOCA/IOB object (<Object name>) is null. Beim Einlesen der Grafik gibt es Probleme während der Dekomprimierung von IOCA and IOB Images. Bitte kontaktieren Sie Maas High Tech Software GmbH und stellen Sie die Error- und Log-Dateien bereit.

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-92	E092: Unsupported image type <ImageType>. Only TIFF, JPEG Image types is supported for AFP Container Objects. Der Grafiktyp im AFP Container-Objekt wird nicht unterstützt. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-94	E094: AFP font representation object for Characerset <CSName> is null. FOCA-spezifischer Fehler Prüfen Sie bitte die AFP Font-Ressource
-95	E095: Used code point information is missing. CharacterSet:<CSName> CodePage:<CPName> Interner Fehler Bitte kontaktieren Sie Maas High Tech Software GmbH.
-96	E096: Error while creating type3 output font representation object for CharacterSet <CSName> and CodePage <CPName>. FOCA-spezifischer Fehler Prüfen Sie bitte die AFP Font-Ressource
-97	E097: Type3 character bitmap array is null for CharacterSet <CSName> and CodePage <CPName>. FOCA-spezifischer Fehler Prüfen Sie bitte die AFP Font-Ressource
-98	E098: Unable to open file <FileName>. Reason: <InfoText> AFP2web kann die angegebene Datei nicht öffnen. Bitte überprüfen Sie die genannte Ursache. Wenn Sie das Problem nicht beheben können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-99	E099: Invalid AFP file. System-interner Fehler. Wenn Sie das Problem nicht beheben können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-100	E100: Memory allocation error : <infotext> Fehlender Speicher. Bitte RAM erweitern.
-101	E101: Invalid INI File Path (<InfoText>) Pfad zur INI-Datei ist nicht gültig. Den angegebene Pfad zur afp2web.ini prüfen.
-102	E102: Error while opening Statistics file <FileName> Pfad für Statistik-Datei ist ungültig. Überprüfen sie die verwendete INI-Datei und die Parameter beim Programmaufruf und stellen Sie sicher, dass der korrekten Pfad für die Statistik-Datei verwendet wird.

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-103	E103: Invalid Output path (<InfoText>). Pfad für die Ausgabe ungültig. Überprüfen sie die verwendete INI-Datei und die Parameter beim Programmaufruf und stellen Sie sicher, dass der korrekten Pfad für die Ausgabe-Dateien (PDF or TIFF) verwendet wird.
-104	E104: INI File (<InfoText>) not found. Die Datei afp2web.ini wird nicht gefunden. Überprüfen Sie ob im angegebenen Pfad die afp2web.ini zu finden ist.
-105	E105: [<SectionName>] section not found in the INI File (<FileName>) Ein erforderlicher Abschnitt fehlt in der INI-Datei. Überprüfen Sie den Inhalt der afp2web.ini.
-106	E106: Type3 Encoding stream is null for CharSet <CSName> and CodePage <CPName>. FOCA-spezifischer Fehler Prüfen Sie bitte die AFP Font-Ressource
-107	E107: Missing or Invalid Spool File Type parameter
-108	E108: Missing Input File parameter Eingabedatei fehlt. Geben Sie mindestens eine Eingabedatei für die Konvertierung an.
-112	E112: <[SectionName]> Section: Missing or Invalid Value <LineText> in mapping.def. Fehlende oder inkorrekte Werte im angegebenen Abschnitt der mapping.def. Bitte überprüfen Sie die Einträge in der mapping.def.
-114	E114: Error while setting font. CharSetName: <CharsetName> Reason: <InfoText> AFP2web konnte die Schrift für die Ausgabe nicht einstellen. Bitte überprüfen Sie die angegebene Ursache für die Fehlermeldung. Wenn Sie das Problem nicht beheben können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-115	E115: Product version expired. Please contact afp2web@maas.de AFP2web Lizenz abgelaufen. Bitte überprüfen Sie die angegebene Ursache für die Fehlermeldung. Wenn Sie das Problem nicht beheben können, kontaktieren Sie bitte Maas High Tech Software GmbH.

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-118	<p>E118: Output font representation is null for CharacterSet=<CSName>, CodePage=<CPName>, OutputFontType=<FontType> Eine oder mehrere externe Schriften sind entweder defekt oder ungültig. Überprüfen Sie bitte die externen Schriften im Ordner "extfont". Wenn Sie das Problem nicht beheben können, kontaktieren Sie Maas High Tech Software GmbH:</p>
-119	<p>E119: Invalid Type1 PFM File <FileName> PFM Datei ist defekt oder ungültig. Bitte überprüfen Sie die angegebene PFM Datei. Wenn Sie das Problem nicht beheben können, kontaktieren Sie bitte Maas High Tech Software GmbH.</p>
-120	<p>E120: Error while parsing True Type font file <FileName>. Reason: <InfoText>. TrueType font ist defekt oder ungültig. Bitte überprüfen Sie die angegebene TrueType Datei. Wenn Sie das Problem nicht beheben können, kontaktieren Sie bitte Maas High Tech Software GmbH.</p>
-155	<p>E155: Error while opening the Index File (<FileName>), leaving... Indexdatei kann nicht mit Schreibzugriff geöffnet werden. Stellen Sie sicher, dass der Pfad zu der Indexdatei vorhanden ist und dass AFP2web Schreibzugriffberechtigung erhält. Wenn Sie das Problem nicht beheben können, kontaktieren Sie bitte Maas High Tech Software GmbH.</p>
-158	<p>E158: File Pointer is null Interner Fehler Bitte kontaktieren Sie Maas High Tech Software GmbH.</p>
-167	<p>E167: Conversion from <InputFileType> to <OutputFileType> is not supported Die Dateikonvertierung wird nicht unterstützt. Keine Aktion erforderlich.</p>
-190	<p>E190: Internet Connection failed. Reason:<InfoText> AFP2web kann keine Session für die Internetverbindung erstellen. Bitte überprüfen Sie die angegebene Ursache für die Fehlermeldung. Wenn Sie das Problem nicht beheben können, kontaktieren Sie bitte Maas High Tech Software GmbH.</p>
-191	<p>E191: FTP Connection failed. Reason:<InfoText> AFP2web kann keine FTP-Verbindung herstellen. Bitte überprüfen Sie die angegebene Ursache für die Fehlermeldung. Wenn Sie das Problem nicht beheben können, kontaktieren Sie bitte Maas High Tech Software GmbH.</p>

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-192	E192: Unable to open FTP file <FileName>. Reason: <InfoText>. AFP2web kann über die FTP-Verbindung eine Datei nicht öffnen. Bitte überprüfen Sie die angegebene Ursache für die Fehlermeldung. Wenn Sie das Problem nicht beheben können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-193	E193: Unable to read FTP file <FileName>. Reason: <InfoText>. AFP2web kann über die FTP-Verbindung eine Datei nicht lesen. Bitte überprüfen Sie die angegebene Ursache für die Fehlermeldung. Wenn Sie das Problem nicht beheben können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-198	E198: Null FTP Connection. Interner Fehler Bitte kontaktieren Sie Maas High Tech Software GmbH.
-199	E199: Reading bytes beyond the bounds.
-200	E200: Null Input Buffer Meldung tritt auf, wenn der Eingabepuffer mit der Länge NULL an die A2WSDK übergeben wird. Bitte stellen Sie sicher, dass der Eingabepuffer mit gültigen Daten an die A2WSDK übergeben wird.
-207	E207: Output file name is null Name der Ausgabedatei fehlt in den Aufrufoptionen. Bitte geben Sie den Namen der Ausgabedatei mit in den Aufrufoptionen.
-231	E231: libXPDF Error (rc=<ErrorCode>): <ErrorMessage> at <API Name> Ein Fehler ist aufgetreten während dem Parsing von PDF Dateien mit ungültigen Einträgen. Überprüfen Sie bitte, ob die PDF-Datei gültig ist und auch im Acrobat Reader angezeigt werden kann.

MHTIMG Library Meldungen

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-1	MIL001: Unknown error. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-2	MIL002: Message id <MsgID> define into ErrorCode enum structure, but Error message missing within achImgLibErrorMsg[]. Check code. Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-5	MIL005: Image header information is null Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-6	MIL006: DIB buffer is null Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-7	MIL007: Error while reading file <FileName>. Rc:<ReturnCode> Msg:<Reason> MHTIMG Library konnte die angegebene Datei nicht lesen. Stellen Sie sicher, das die MHTIMG Lib Lesezugriff auf die Datei hat. Wenn Sie das Problem nicht lösen können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-8	MIL008: File name parameter is null or empty Dateiname an MHTIMG Library ist null Geben Sie die korrekte Dateiname an die MHTIMG Library
-9	MIL009: Unsupported image type <ImageType> MHTIMG Library unterstützt nicht den angegebene Grafiktyp. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-10	MIL010: Decompression of <BPP> Bits/Pixel <Comp.Type> <ImageType> images is not supported Die Dekomprimierung wird nicht unterstützt. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-12	MIL012: Source image buffer is null or buffer length parameter is zero Es wurden ungültige Parameter an die MHTIMG Library übergeben. Stellen Sie sicher, dass die korrekte Pufferbereich für die Grafik an MHTIMG Library übergeben wird. Wenn Sie das Problem nicht lösen können, kontaktieren Sie bitte Maas High Tech Software GmbH.

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-13	MIL013: Invalid compression parameter id <ParamId> Ungültiger Parameter für die Komprimierung an MHTIMG Library Übergeben sie korrekte Parameter an die MHTIMG Library
-14	MIL014: IOCA Library handle is null Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-15	MIL015: IOCA Library Error: Rc:<ReturnCode> Msg:<InfoText> Fehler in der IOCALib Library. Überprüfen Sie bitte die Information der Fehlermeldung. Wenn Sie das Problem nicht lösen können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-16	MIL016: Decompression of <SPP> Samples/Pixel <ImageType> images is not supported Diese Dekomprimierung wird nicht unterstützt. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-17	MIL017: CxImage Library Error: <InfoText> Ein Fehler ist aufgetreten in der CxImage Library. Prüfen Sie die eingelesene Grafikdatei bzw. -datenstrom anhand der Fehlermeldung. Wenn Sie das Problem nicht lösen können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-18	MIL018: Error while opening file <FileName>. Rc:<ReturnCode> Msg:<Reason> MHTIMG Library konnte die Datei nicht öffnen. Stellen Sie sicher, dass die Datei vorhanden ist.
-19	MIL019: Error while creating TIFF handle for <Filename> "Memory Stream"> Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-20	MIL020: TIFF Library Error: <InfoText> Fehler in der TIFF Library. Überprüfen Sie bitte die Informationen in der Fehlermeldung. Wenn Sie das Problem nicht lösen können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-21	MIL021: Memory allocation error Unzureichender Speicher Erhöhen Sie den RAM.
-22	MIL022: Compression <Compression> is not supported for <OutputFormat> Die Komprimierung wird für das angegebene Ausgabeformat nicht unterstützt. Bitte kontaktieren Sie Maas High Tech Software GmbH.

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-23	MIL023: <ImageType> Image encoder is null Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-24	MIL024: <ImageType> Image decoder is null Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-25	MIL025: JPEG compression failed: <InfoText> Programmfehler. Überprüfen Sie bitte die Informationen in der Meldung. Wenn Sie das Problem nicht lösen können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-26	MIL026: <MethodName> not implemented for <ClassName> Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-27	MIL027: JPEG Library handle is null Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-28	MIL028: Compression parameter array is null Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-29	MIL029: Cannot convert IOCA <Colorspace> image planes to <Colorspace> color space Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-30	MIL030: CMKY image buffer passed to CMYK Converter is null Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-31	MIL031: IOCA RGB image plane buffer is null
-32	MIL032: Unsupported file format <FileType> MHTIMG Library unterstützt nicht die Dateityp. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-33	MIL033: Grayscale conversion of<Bits/Pixel> Bits/Pixel images is not supported

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-34	MIL034: Black/White conversion of <Bits/pixel> Bits/Pixel images is not supported
-37	MIL037: IOCA logical resolution unit not supported MHTIMG Library unterstützt nicht die IOCA Logical Auflösungseinheit. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-38	MIL038: Decompression of <Comp.Type> <ImageType> images is not supported Die Dekomprimierung wird nicht unterstützt. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-39	MIL039: Error while writing file <FileName> MHTIMG Library konnte die Datei nicht schreiben. Stellen Sie sicher, dass die MHTIMG Library einen Lesezugriff für den angegebenen Pfad erhält. Wenn Sie das Problem nicht lösen können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-40	MIL040: Incorrect call to buffer access methods Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-42	MIL042: Unable to prepare header for OJPEG TIFF data Die TIFF Datei oder der TIFF Grafikpuffer ist entweder ungültig oder defekt. Überprüfen Sie den TIFF Header der TIFF Datei. Wenn Sie das Problem nicht lösen können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-43	MIL043: Error while extracting TIFF image data from strip <StripNumber> Die TIFF Datei oder der TIFF Grafikpuffer ist entweder ungültig oder defekt. Überprüfen Sie bitte die TIFF Datei. Wenn Sie das Problem nicht lösen können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-44	MIL044: JPEG Library Error: <InfoText> Fehler in der JPEG Library Überprüfen Sie bitte die Information in der Fehlermeldung. Wenn Sie das Problem nicht lösen können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-45	MIL045: Images with Bits/Pixel greater than one can not converted to <ImageType> Die Konvertierung wird nicht unterstützt. Bitte kontaktieren Sie Maas High Tech Software GmbH.

<i>Rückgabewert</i>	<i>Meldetext und Beschreibung</i>
-46	MIL046: TIFF Library handle is null Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-47	MIL047: Error while writing <SFIType> SFI Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-48	MIL048: Image data is null Programmfehler. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-50	MIL050: <ImageType> compression is not supported MHTIMG Library unterstützt nicht die Komprimierung. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-51	MIL051: Error while decompressing IOCA <Colorspace> planes of image data Ungültige IOCA Grafikdaten. Prüfen Sie, ob die IOCA Grafik gültige Daten enthält. Wenn Sie das Problem nicht lösen können, kontaktieren Sie bitte Maas High Tech Software GmbH.
-52	MIL052: Decompression of <BPP> Bits/Pixel <ImageType> images is not supported. Die geforderte Dekomprimierung wird nicht unterstützt. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-53	MIL053: Decompression of <ImageType> images having <PhotoMetric-Name> Photometric Interpretation is not supported Die geforderte Dekomprimierung wird nicht unterstützt. Bitte kontaktieren Sie Maas High Tech Software GmbH.
-55	MIL055: Insufficient data read from <StreamName>. Expected-Bytes=<ScanlineSize> ReadBytes=<BytesReadFromStream> Der Grafikdatenstrom hat weniger Bytes zur Verfügung als für eine komplett eingelesene Bildzeile benötigt werden. Prüfen Sie bitte, ob die Grafikdatei oder -datenstrom gültige Daten enthält.
-57	MIL057: TIFF Images with Planar format(separate planes of data) are not supported TIFF Grafiken im Planar Format (mit Bitplanes) werden nicht unterstützt. Bitte kontaktieren Sie Maas High Tech Software GmbH.

Kapitel A: Anhang

Dieser Anhang enthält:

- Tipps für die Migration der Scripting Facility Module zu AFP2web Version 3.x
- Anleitung zu den Scripting Facility Beispielen
- Frequently Asked Questions

A.1 Migration der Scripting Facility Module zu Version 3.x

Die folgende Tabelle zeigt die Unterschiede zwischen den Scripting Modulen der vorigen AFP2web Version 2.1 zu den von AFP2web Version 3.x. Sie können diese Auflistung als eine Checkliste für die Migration verwenden.

<i>Version 2.1</i>	<i>Version 3.x</i>
use a2w::mhtIni;	use a2w::Config;
use a2w::mhtDocument;	use a2w::Document;
use a2w::mhtPtocaPage;	use a2w::Page;
use a2w::mhtTextObject;	use a2w::Text;
use a2w::mhtNOP;	use a2w::NOP;
use a2w::mhtResOverlay;	use a2w::Overlay;
use a2w::mhtAttributeElement;	use a2w::Index;
use a2w::mhtAFPFont;	use a2w::Font;
use a2w::mhtPagePageGroupIndex;	use a2w::Index;
use a2w::mhtIndexElement;	use a2w::Index;
sub initialize(mhtIni)	sub initialize(a2w::Config, a2w::Kernel)
\$a2wIniPar->setAutosplit(\$TRUE);	\$a2wConfigPar->setAttribute("Auto-split", "on");
\$svIndexPathTmp = \$a2wConfig->getIndexPath();	\$svIndexPathTmp = \$a2wConfigPar->getAttribute("IndexPath");
\$svOutputFilePathTmp = \$a2wConfig->getOutputFilePath();	\$svOutputFilePathTmp = \$a2wConfigPar->getAttribute("OutputFilePath");
\$svScriptArgsTmp = \$a2wConfig->getScriptArgs();	\$svScriptArgsTmp = \$a2wConfigPar->getAttribute("ScriptArgument");
\$a2wPageTmp = \$a2wDocument->getFirstPageObject();	\$a2wPageTmp = \$a2wDocumentPar->getFirstPage();
\$a2wPageTmp = \$a2wDocument->getNextPageObject();	\$a2wPageTmp = \$a2wDocumentPar->getNextPage();

<i>Version 2.1</i>	<i>Version 3.x</i>
<code>\$a2wPage->getFirstTextObject();</code>	<code>\$a2wPagePar->getFirstText();</code>
<code>\$a2wPage->getNextTextObject();</code>	<code>\$a2wPagePar->getNextText();</code>
<code>\$a2wPageGroupIndexTmp = \$a2wDocument->getFirstIndexObject();</code>	<code>\$a2wIndexTmp = \$a2wDocumentPar-> getFirstIndex();</code>
<code>\$a2wPageGroupIndexTmp = \$a2wDocument->getNextIndexObject();</code>	<code>\$a2wIndexTmp = \$a2wDocumentPar-> getNextIndex();</code>
<code>\$a2wIndexTmp = \$a2wPageGroupIndexTmp->getFirstAttri- buteElement();</code>	directly access to index thru: <code>\$a2wIndexTmp = \$a2wDocumentPar-> getFirstIndex();</code> or <code>\$a2wIndexTmp = \$a2wPagePar->getFirstIndex();</code>
<code>\$a2wIndexTmp = \$a2wPageGroupIndexTmp->getNextAttri- buteElement();</code>	directly access to index thru: <code>\$a2wIndexTmp = \$a2wDocumentPar-> getNextIndex();</code> or <code>\$a2wIndexTmp = \$a2wPagePar->getNextIndex();</code>
<code>\$a2wIndex->setAttributeName("Insured");</code>	<code>\$a2wIndex->setName("Insured");</code>
<code>\$a2wIndex->setAttributeValue("John Doe");</code>	<code>\$a2wIndex->setValue("John Doe")</code>
<code>\$svName = \$a2wIndex->getAttribute- Name();</code>	<code>\$svName = \$a2wIndex->getName();</code>
<code>\$svValue = \$a2wIndex->getAttributeVa- lue();</code>	<code>\$svValue = \$a2wIndex->getValue()</code>
<code>\$sIndexNameTmp = \$a2wIndexTmp-> getAttributeName();</code>	<code>\$sIndexNameTmp = \$a2wIndexTmp->get- Name();</code>
<code>\$sIndexValueTmp = \$a2wIndexTmp-> getAttributeValue();</code>	<code>\$sIndexValueTmp = \$a2wIndexTmp->get- Value();</code>
<code>\$a2wOverlay->getFirstTextObject();</code>	<code>\$a2wOverlay->getFirstText();</code>
<code>\$a2wOverlay->getNextTextObject();</code>	<code>\$a2wOverlay->getNextText();</code>

<i>Version 2.1</i>	<i>Version 3.x</i>
<code>\$a2wDocument->addIndex("Insured", "John Doe");</code>	<code>\$a2wIndexTmp = new a2w::Index();</code>
	<code>\$a2wIndexTmp->setName("Insured");</code>
	<code>\$a2wIndexTmp->setValue("John Doe");</code>
	<code>\$a2wDocumentPar->addIndex(\$a2wIndexTmp);</code>
<code>\$a2wPage->addIndex("Insured", "John Doe");</code>	<code>\$a2wIndexTmp = new a2w::Index();</code>
	<code>\$a2wIndexTmp->setName("Insured");</code>
	<code>\$a2wIndexTmp->setValue("John Doe");</code>
	<code>\$a2wPagePar->addIndex(\$a2wIndexTmp);</code>
<code>\$a2wPage->includeObject(0x92, "BGFORM", \$XPos, \$YPos, \$Rotation, "JPEG");</code>	<code>\$a2wPagePar->addImage("BGFORM", \$XPos, \$YPos, \$svWidth, \$svHeight, \$Rotation);</code>
<code>\$a2wPage->includeObject(0xDF, "O1BGFORM", \$XPos, \$YPos, \$Rotation);</code>	<code>\$a2wPagePar->addOverlay("O1BGFORM", \$XPos, \$YPos);</code>
<code>\$a2wPage->includeObject(0x5F, "S1BGFORM", \$XPos, \$YPos, \$Rotation);</code>	<code>\$a2wPagePar->addPSEG("S1BGFORM", \$XPos, \$YPos);</code>
<code>\$a2wPage->addObject(\$a2wTextTmp);</code>	<code>\$a2wPagePar->addText(\$a2wTextTmp);</code>
<code>\$a2wTextTmp->setColor(0x00BBGRR);</code>	<code>\$a2wTextTmp->setColor(0x00RRGGBB);</code>

A.2 Scripting Facility Beispiele

Einführung

Dieser Teil dient als Einführung in die Verwendung der Scripting Facility Beispiele, die Sie mit der Installation von AFP2web Version 3 erhalten. Jedes Beispiel zeigt exemplarisch die Programmierung für eine kleine Aufgabe. Die Beispiele enthalten Kommentare, die das Verständnis erleichtern.

Umgebung für die Ausführung der Scripting Facility Beispiele

AFP2web Version 3.x ist schnell installiert. Welche Systemvoraussetzungen für AFP2web generell erforderlich sind und wie Sie AFP2web Version 3.x installieren werden im Benutzerhandbuch, in dem Kapitel für die Installation, ausführlich beschrieben.

Die Scripting Facility Beispiele sind in Perl programmiert. Sie finden diese als Beispieldateien mit der Dateinamenserweiterung *.pm im Installationsverzeichnis von AFP2web.

Hinweis: Es ist nicht erforderlich, Perl für die Beispiele zu installieren. Eine Perl Engine wird als Perl DLL mit AFP2web geliefert und automatisch ausgeführt, wenn diese DLL im aktuellen Verzeichnis oder über die Umgebungsvariable PERLLIB gefunden wird. Wenn Sie Ihre eigene Perl Umgebung verwenden, beachten Sie bitte die Systemvoraussetzungen im Kapitel über die Installation von AFP2web.

AFP2web ermittelt seine Laufzeitparameter aus der Datei afp2web.ini Datei (INI Datei). Jede Kommandozeilenoption, die eingegeben wird, ersetzt den entsprechenden Parameter in der INI Datei. Wenn Logging aktiviert ist, erfolgt standardmäßig die Ausgabe von Log-Meldungen in den Unterordner /pdf. Eine detaillierte Beschreibung der Parameter und Kommandozeilenoptionen finden Sie ebenfalls im Referenzteil des Benutzerhandbuchs zu AFP2web.

Ausführung eines Scripting Facility Beispiels

Aufruf mittels der Demo Batch Befehlsdatei

Jedes Perl Modul enthält zu Beginn einen Block-Kommentar, der beschreibt, wie Sie AFP2web für die Ausführung des Beispiels aufrufen. Ein Beispiel für den Aufruf von sortPageTextObjs.pm:

Listing

Demo.bat (AFP2web starten und ein Skript ausführen)

```
On Windows:
afp2web.exe -q -c -doc_cold -sp: sortPageTextObjs.pm samples\insure.afp

On Unix:
./afp2web -q -c -doc_cold -sp: sortPageTextObjs.pm samples/insure.afp
```

Die verwendeten Kommandozeilenoptionen und deren Beschreibung:

<i>Parameter:</i>	<i>Beschreibung</i>
-q	AFP2web in Quiet Modus ausführen (Meldungen an die Konsole unterdrücken).
-doc_cold	Die AFP2web Scripting Facility aktivieren.
-sp	Name der auszuführenden Skript-Datei, wenn dieser nicht afp2web.pm lautet.
-sa	Argumente, die an das Skript übergeben werden.

Weitere INI Parameter und Kommandozeilenoptionen finden Sie im Referenzteil des Benutzerhandbuchs zu AFP2web. Beispiel: Dort finden Sie die Beschreibung der Option -ll, mit der Sie ein Verarbeitungsprotokoll erzeugen.

Die folgenden Zeilen zeigen, wie Sie im Fehlerfall den Rückgabewert abfangen und in der Kommandozeile anzeigen.

Listing
Demo Batch Datei (Fehlerfall anzeigen)

```
IF %ERRORLEVEL% EQU 0 (
@ECHO AFP2web: Done %ERRORLEVEL%
) ELSE (
@ECHO AFP2web: Error %ERRORLEVEL%
)
```

Das Skript mittels demo.bat starten.

<i>Schritt</i>	<i>Beschreibung</i>
1	Editieren Sie die Batch-Befehlsdatei demo.bat, bzw. demo und geben Sie die erforderlichen Befehle ein für die Ausführung des erwünschten Skripts.
	Um das Skript-Beispiel auszuführen starten Sie die Demo Stapeldatei. Unter Windows doppelklicken Sie auf die Datei im Windows Explorer. Das Fenster für die Kommandozeile wird geöffnet, AFP2web wird zur Ausführung gebracht. Meldungen an der Konsole weisen auf den Ablauf der Bearbeitung hin. Wurde die Ausgabe von Log-Meldungen für AFP2web aktiviert, dann schreibt AFP2web Log-Meldungen standardmäßig in eine Datei im Unterverzeichnis /log.
2	Sichten Sie die Ergebnisse.

Überblick der Skript Beispiele

Die folgende Tabelle gibt zu jedem Skript Beispiel eine Kurzbeschreibung:

<i>Skript Beispiel</i>	<i>Beschreibung</i>
dumpIniParms.pm	dumpIniParms.pm schreibt die Einstellungen aller INI Parameter in eine Ausgabedatei. (Ist eine Kommandozeilenoption angegeben, so ersetzt diese Option den entsprechenden INI Parameter.)
autosplit.pm	autosplit.pm zeigt wie man die AutoSplit Funktionalität aktiviert. Autosplit weist AFP2web an, eine AFP Spooldatei automatisch in einzelne Dokumente und Dokumentseiten zu zerlegen. Dies ist jedoch nur dann möglich, wenn die AFP Spooldatei die AFP-spezifische Page Groups und Tag Logical Elements (TLEs) enthält. Das Skript zeigt auch, wie man die in den TLEs definierten Indexdaten ausgibt.
dumpPageTextObjs.pm	dumpPageTextObjs.pm zeigt, wie man auf die Textobjekte auf einer Seite zugreift und die Texte in eine (Dump-) Datei ausgibt. Bei jeder Änderung der Schriftattribute wird diese Information zuvor mit ausgegeben.
sortPageTextObjs.pm	In den AFP Druckdaten entspricht die Reihenfolge der Textobjekte nicht zwingend der Reihenfolge, in der diese auf einer Druckseite ausgegeben werden. Dies erschwert das Lesen der AFP Daten in einer Dump. sortPageTextObjs.pm zeigt wie man alle Textobjekte auf einer Seite nach deren X/Y Koordinatenpositionen sortiert und in dieser Reihenfolge ausgibt.
dumpMediumMaps.pm	dumpMediumMaps.pm erstellt eine Auflistung aller Medium Maps einer Dokumentenseite.
dumpNOPs.pm	NOP Felder werden verwendet, um einen Datenstrom mit nicht sichtbaren Dokumentinformationen anzureichern. Unterschiedliche Anwendungen haben ihre eigenen Konventionen für NOP Felder und verwenden diese beispielsweise für Indizierung, Archivierung und für die Verarbeitungssteuerung. dumpNOPs.pm erstellt eine Auflistung der Werte, die in den NOP-Feldern einer Seite mitgegeben werden.
eyecatcher.pm	eyecatcher.pm findet und extrahiert vordefinierte Zeichenfolgen im Text einer Seite. Jede Zeichenfolge hat eine vorgegebene X/Y Koordinatenposition. Dieses Skriptbeispiel erkennt diese Zeichenfolgen entweder als Signal für den Beginn eines neuen Dokuments oder als eine Stelle zum Herauslesen von Indexdaten aus dem Dokumenteninhalt.
addBookmarks.pm	addBookmarks.pm erstellt Lesezeichen in einem PDF Dokument.

<i>Skript Beispiel</i>	<i>Beschreibung</i>
addWatermark.pm	Dieses Skriptbeispiel fügt einer Seite eine Hintergrundgrafik als Wasserzeichen hinzu.
addAnnotations.pm	addAnnotations.pm erstellt eine Hyperlink-Verknüpfung im PDF Dokument.
.pm	Dieses Beispiel liest den Inhalt eines AFP Dokuments und gibt Informationen über den Struktur und Inhalt des Dokuments im XML Format aus. Dieses Skript benötigt eine vorinstallierte Perl Laufzeitumgebung.

Die Grundlegende Struktur eines Skripts

Wenn Sie ein neues Skriptmodul für die Scripting Facility erstellen, verwenden Sie bitte die Mustervorlage afp2web.pm. Hier sind bereits grundlegende Funktionen implementiert (wie z.B. das Einfügen von Perl Packages, das Einlesen von Parametern, die Ausgabe von Log-Meldungen, die Ausgabe von Indexdaten und die Zerlegung eines AFP Spools in Dokumente, bzw. in Seiten).

Ein Skript muss die folgenden Routinen beinhalten:

- sub afp2web()
- sub initialize()
- sub initializeDoc()
- sub initializePage()
- sub finalizePage()
- sub finalizeDoc()
- sub finalize()

Der Zweck einer Routine ist es, bestimmte Verarbeitungsereignisse von AFP2web abzufangen.

<i>AFP2web Kernel</i>		<i>Unterroutine in afp2web.pm</i>
Prozess Beginn		initialize() (Only once)
Dokumentbeginn		initializeDoc()
Die folgenden Unter Routinen werden in Folge als einen logischen Schritt für eine Dokumenten-seite ausgeführt:		

AFP2web Kernel		Unterroutine in <i>afp2web.pm</i>
Seitenbeginn		initializePage()
Seite ist geparkt		afp2web()
Seite ist verarbeitet		finalizePage()
Dokument ist verarbeitet		finalizeDoc()
Prozessende		finalize (Only once)

Jede Unterroutine erhält Eingabe-Argumente, die Sie als Referenz auf Objekte zunächst sichern und später in einer Folge-Routine verarbeiten. Weitere Informationen hierzu finden Sie in der Beschreibung der Scripting Facility des AFP2web Benutzerhandbuchs.

Ein Skript beginnt in der Regel mit der Einbettung von Packages mit der Scripting Facility Funktionalität. Ein Beispiel:

Listing
dumplniParms.pm - Packages

```
use a2w: : Conf i g;
use a2w: : Kernel ;
```

Weitere Packages können ebenfalls eingebunden und ermöglichen so die Erweiterung des Skripts um weitere Funktionen.

Wichtig:	
	In den Beschreibungen der einzelnen Skriptmodule beschränken wir uns auf die wesentlichen Subroutinen. Weitere Informationen finden Sie in den Inline-Kommentaren des Skriptmoduls.

dumpIniParams.pm: INI Parameters ausgeben

Das Script Facility Beispiel `dumpIniParams.pm` zeigt, wie man auf die AFP2web Laufzeitparameter zugreift. Die Werte der Parameter sind in der Konfigurationsdatei (`afp2web.ini`) festgelegt oder werden von einer Programmaufrufoption überschrieben.

dumpIniParams.pm verstehen

Das Skript erfordert die folgenden Packages:

Listing <i>dumpIniParams.pm - Packages</i>
<pre>use a2w: : Config; use a2w: : Kernel ;</pre>

Die Subroutine `initialize()` wird zu Beginn des AFP2web Prozesses aufgerufen. Zu diesem Zeitpunkt haben wir Zugang zu den globalen Parametern. Dieser werden als Argumente an die Subroutine übergeben. Wir speichern diese Argumente in der Perl Variable `@_`.

Listing <i>dumpIniParams.pm - sub initialize() - Argumente</i>
<pre>#---- Get Parameter of initialize(Par: a2w: : Config, a2w: : Kernel) (\$a2wConfigPar, \$a2wKernelPar) = @_;</pre>

Die folgende Anweisung aktiviert die Ausgabe von Log-Meldungen, wenn der entsprechende INI Parameter oder Kommandozeilenoption gesetzt ist.

Listing***dumpIniParms.pm - sub initialize() - Log-Meldungen aktivieren, wenn erwünscht***

```
#---- Set/Reset Logging
$bLog = $FALSE;
if (index( lc($a2wConfigPar->getAttribute("LoggingLevel")), "sf") >= 0 ){
    $bLog = $TRUE;
}
```

In diesem Kontext haben wir Zugriff auf die Funktionen der Packages `a2w::Config` und `a2w::Kernel`. Um beispielsweise einen beliebigen Parameter zu lesen, verwenden wir die `getAttribute` Methode von `a2w::Config` und übergeben den Namen des INI Parameters als Argument. Die folgenden Zeilen zeigen den Aufruf:

Listing***dumpIniParms.pm - sub initialize() - Methoden von a2w::Config***

```
my $svScriptProcTmp = $a2wConfigPar->getAttribute("ScriptProcedure");
my $svScriptArgsTmp = $a2wConfigPar->getScriptArgs();
my $svIndexPath = $a2wConfigPar->getIndexFilePath();
my $svOutputFilePath = $a2wConfigPar->getOutputFilePath();
my $svSpoolFilename = $a2wKernelPar->getSpoolFilename();
```

Die folgende Zeilen zeigen die Anweisungen für die Ausgabe in eine separate Datei. Der Name dieser Datei muss beim Aufruf von AFP2web als Kommandozeilenoption `-sa` angegeben sein. Fehlt diese Option, so reicht das Skript einen negativen Rückgabewert an AFP2web und AFP2web beendet die Verarbeitung.

Listing***dumpIniParms.pm - sub initialize() - Separate Ausgabedatei für das Skript***

```
#---- Open Dump file
my ($svSpoolFilenamePathTmp, $svDumpFilenameTmp) = ($svSpoolFilename =~
/^(?\. *[:\\\/])?(.*)/s);
$svDumpFilenameTmp = $svOutputFilePath . $svDumpFilenameTmp . ".txt";
open( fDumpFile, ">$svDumpFilenameTmp" );
print "Running $svScriptProcTmp: Dumping to $svDumpFilenameTmp...\n";
```

Die folgenden Zeilen erstellen ein Array zur Aufnahme aller AFP2web Parameter. In einer Schleife werden die Werte in die Datei ausgegeben.

Listing
dumplniParams.pm - sub initialize() - Parameter ausgeben

```
my $ptrTmp = 0;
my @iniParmLi st = ();

@ini ParmLi st[$ptrTmp++] = "Li censee";
@ini ParmLi st[$ptrTmp++] = "Serial Nr";
@ini ParmLi st[$ptrTmp++] = "Ti tle";
@ini ParmLi st[$ptrTmp++] = "Subj ect";
@ini ParmLi st[$ptrTmp++] = "Keywords";
# ----- additional i tems not listed here...

$ini ParmCountTmp = @ini ParmLi st;

print fDumpFile ("===== Ini Parameter List =====\n");
for ( $ptrTmp = 0; $ptrTmp < $ini ParmCountTmp; $ptrTmp++ ){
print fDumpFile ( "\t" . @ini ParmLi st[$ptrTmp] . "=>" . $a2wConfi gPar-
>getAttri bute(@ini ParmLi st[$ptrTmp]) . "<=\n" );

}
print fDumpFile ("===== \n");
close(fDumpFi le);
return 0;
```

Konvertierte Ausgabedokumente werden nicht ausgegeben. Dies deshalb, weil wir in der Skript Subroutine `afp2web()` für jede Dokumentseite den Wert `$SKIP` zurückgeben.

Listing
dumplniParams.pm - sub afp2web()

```
sub afp2web(){

if ( $bLog == $TRUE ){
print "afp2web(): PageId " . $a2wPagePar->getParseId() . "\n";
}

$APPEND= 0; # append page to Current Document
$SKIP= 1; # skip page
$NEWDOC= 2; # new document

$svRetTmp = $SKIP; # skip page

return $svRetTmp;
}
```

Das Script Beispiel ausführen

Mit den folgenden Befehlen in der Demo Stapeldatei starten Sie AFP2web mit der Scripting Facility:

Listing **Demo Stapeldatei**

```
@ECHO OFF
: On Windows:
afp2web.exe -q -c -doc_col d -sp: dumpIniParms.pm samples\insure.afp

: On Unix:
./afp2web -q -c -doc_col d -sp: dumpIniParms.pm samples/i nsure.afp
```

Ergebnisse von dumpIniParms.pm

Die Ausgabedatei finden Sie im Zielverzeichnis (standardmäßig ist dies die dumpIniParms.txt in /pdf):

Listing **dumpIniParms.pm - Inhalte der Ausgabedatei**

```
===== Ini Parameter List
=====
License=>Maas High Tech Software GmbH MMHHTT SF<=
SerialNr=>7CDEE0C4-542D980C<=
Title=>AFP2Web Version 3.0 [Built for Windows NT/2000/2003 on Aug 5 2005
at 12: 07: 32]<=
Subject=>AFP and TIFF Conversions (afp2web.com)<=
Keywords=>AFP LPD PDF TIFF<=
Logging=>off<=
LoggingFont=>off<=
LoggingLevel=>0<=
ExceptionLoggingLevel=>1<=
LogPath=>./log/<=
ResPath=>./samples/resource/<=
CpPath=>./afpcp/<=
CodePage=>T1V10273<=
OutputFilePath=>./pdf/<=
IndexPath=>./pdf/<=

.... additional lines not displayed here....
```

autosplit.pm: Eine AFP Spooldatei in Dokumente und Seiten trennen

AFP Dokumente sind üblicherweise für die Druck-Ausgabe in Spooldateien gesammelt. Eine AFP Spooldatei kann eine Vielzahl an Dokumenten und Seiten beinhalten. Zu den nicht sichtbaren Informationen in der Spooldatei gehören auch Indexdaten, die in AFP-spezifischen Tagged Logical Elements (TLEs) definiert sind. TLEs sind logisch auf Dokumente und Dokumentseiten bezogen und zeigen AFP2web die Dokument- und Seitengrenzen an.

Wenn Ihr Typ von AFP Spooldatei TLEs beinhaltet, so können Sie AFP2web die Trennung in Dokumente und Seiten automatisch ausführen lassen. Hierzu aktivieren Sie mit der Scripting Facility die AutoSplit Funktion. Gleichzeitig können Sie die Scripting Facility während der Dokumentkonvertierung nutzen und andere Aufgaben erledigen. Sie aktivieren die AutoSplit Funktion indem Sie den Parameter AutoSplit auf ON setzen. AFP2web trennt dann die AFP Spooldatei in Dokumente und Seiten.

Das Skript-Beispiel *autosplit.pm* zeigt wie Sie die AutoSplit Funktion von AFP2web nutzen.

autosplit.pm verstehen

In dieser Beschreibung beschränken wir uns auf die folgenden Subroutinen:

<i>Subroutine</i>	<i>Beschreibung</i>
sub afp2web()	Einstieg pro Dokumentseite. Der Aufruf geschieht durch AFP2web. Der Rückgabewert weist AFP2web an, ein neues Dokument für die Seite zu beginnen, die Seite zu ignorieren, oder die Verarbeitung an dieser Stelle komplett abubrechen.
sub initialize()	Zugriff auf Verarbeitungsparameter, aktiviert Autosplit und initialisiert Variablen.
sub initializeDoc()	Dokumentobjekt sichern, Seitenzählung neu beginnen
initializePage()	Objekt für die Dokumentseite sichern
addDocumentIndexes()	Indexdaten sammeln
addPageIndexes()	Indexdaten sammeln
sub finalizeDoc()	Indexdaten ausgeben

Die Subroutine `initialize()` wird einmal zu Beginn des AFP2web Prozesses aufgerufen. Hier werden die Parameter gelesen, die AutoSplit Funktion aktiviert und Variablen initialisiert.

Listing
autosplit.pm - initialize() - Autosplit Aktivierung und die Initialisierung von Variablen

```
...

#---- Reset Page I d
$pageId = 0;

#---- Reset/Create Current Index List
@indexList = ();

#---- Set AutoSplit to true
$a2wConfigPar->setAttribute( "AutoSplit", "on" );

...
```

AFP2web startet die Subroutine `initializeDoc()` sobald es mit der Verarbeitung eines neuen Dokuments beginnt. In dieser Routine initialisieren wir die Seitenzählung. Wichtig ist es, in der Variablen `$a2wDocumentPar` eine Referenz auf das aktuelle Dokument zu sichern, um später auf die Objekte des Dokuments zugreifen zu können. Da wir zu diesem Zeitpunkt bereits auf die Indexdaten zum Dokument zugreifen können, rufen wir die Subroutine `addDocumentIndexes()` auf, um die Indexdaten zu sammeln.

Listing***autosplit.pm - initializeDoc() - Dokumentobjekt sichern, Seitenzählung neu beginnen***

```
sub initializeDoc(){  
  
#---- Get Parameter of initializeDoc( Par: a2w::Document )  
($a2wDocumentPar) = @_;  
  
if ( $bLog == $TRUE ){  
print "initializeDoc(): DocId " . $a2wDocumentPar->getId() . "\n";  
}  
  
#---- Reset Page Id  
$PageId = 0;  
  
#---- Add Document Indexes to Index List  
addDocumentIndexes();  
  
return 0;  
}
```

In der Subroutine `initializePage()`, sichern wir in der Variable `$a2wPagePar` eine Referenz auf die aktuelle Seite. Mit dieser Referenz können wir später auf die Indexdaten der Seite zugreifen.

Listing***autosplit.pm - initializePage() - Objekt für die Dokumentseite sichern***

```
sub initializePage(){  
  
#---- Get Parameter of initializePage( Par: a2w::Page )  
($a2wPagePar) = @_;  
  
if ( $bLog == $TRUE ){  
print "initializePage()\n";  
}  
return 0;  
}
```

Die Subroutine `afp2web()` ist der Haupteinstieg für AFP2web und wird für jede Dokumentseite aufgerufen.

Da AFP2web mit der Autosplit Funktion, die Trennung in Dokumente und Seiten selber erledigt, müssen wir für diese Trennung nicht viel tun. Wir müssen beispielsweise keine Regeln definieren für das Erkennen des Beginns eines Dokuments.

Daher geben wir als Rückgabewert stets den Standardwert (aktuelle Seite dem Dokument hinzufügen) und wir kümmern uns lediglich um die Seitenzählung.

Als letzte Aktion sammeln wir die Indexdaten für diese Seite durch Aufruf der Subroutine `addPageIndexes()`.

Listing
autosplit.pm - sub afp2web()

```
sub afp2web(){  
  
    if ( $bLog == $TRUE ){  
        print "afp2web(): PageId " . $a2wPagePar->getParseId() . "\n";  
    }  
  
    $APPEND= 0; # append page to Current Document  
    $SKIP= 1; # skip page  
    $NEWDOC= 2; # new document  
  
    #---- Set default return value  
    my $svRetTmp = $APPEND; # default: append page  
  
    #---- Increment Page Id  
    $PageId++;  
  
    #---- Add Page Indexes to Index List  
    addPageIndexes();  
  
    return $svRetTmp;  
}
```

Es folgen einige Zeilen aus den UnterROUTINEN, welche die Methoden des Dokument- bzw. des Seitenobjekts nutzen um die Indexdaten zu sammeln.

Listing**autosplit.pm - addDocumentIndexes() -**

```
sub addDocumentIndexes(){  
  
    if ( $bLog == $TRUE ){  
        print "addDocumentIndexes()\n";  
    }  
  
    #---- Define temp variables  
    my $IndexPtrTmp = @IndexList;  
  
    #---- Fetch first Document Index  
    my $a2wIndexTmp = $a2wDocumentPar->getFirstIndex();  
  
    #---- Add PageGroup Name  
    if ( $a2wIndexTmp != 0 ){  
  
        #---- Add PageGroup Name to Index List  
        @IndexList[$IndexPtrTmp++] = "PageGroup=" . $a2wIndexTmp->getIndexObject  
            Name();  
    }  
  
    #---- Loop thru Indexes  
    while ( $a2wIndexTmp != 0 ){  
  
        #---- Add Index Record to Index List  
        @IndexList[$IndexPtrTmp++] = $a2wIndexTmp->getName() . "=" .  
            $a2wIndexTmp->getValue();  
  
        #---- Fetch next Document Index  
        $a2wIndexTmp = $a2wDocumentPar->getNextIndex();  
    }  
}
```

Listing
autosplit.pm - addPageIndexes() -

```
sub addPageIndexes(){  
    if ( $bLog == $TRUE ){  
        print "addPageIndexes()\n";  
    }  
  
    #---- Define temp variables  
    my $IndexPtrTmp = @IndexList;  
  
    #---- Add Page Id to Index List  
    @IndexList[$IndexPtrTmp++] = "Page=" . $a2wPagePar->getName();  
  
    #---- Fetch first Page Index  
    my $a2wIndexTmp = $a2wPagePar->getFirstIndex();  
  
    #---- Loop thru Indexes  
    while ( $a2wIndexTmp != 0 ){  
  
        #---- Add Index Record to Index List  
        @IndexList[$IndexPtrTmp++] = $a2wIndexTmp->getName() . "=" .  
            $a2wIndexTmp->getValue();  
  
        #---- Fetch next Page Index  
        $a2wIndexTmp = $a2wPagePar->getNextIndex();  
    }  
}
```

Am Ende eines Dokuments können wir in der Subroutine `finalizeDoc()` die Indexdaten, die wir für das Dokument bisher gesammelt haben, ausgeben. Das folgende Listing zeigt einige Zeilen aus dem Skript. Für die vollständige Programmierung bitten wir Sie das Scriptmodul zu lesen.

Listing***autosplit.pm - sub finalizeDoc() - Indexdaten ausgeben***

```
#---- Open Index file
my $svFileOpenSuccessTmp = open( fIndexFile, ">$IndexFileNameTmp" );

if ( $svFileOpenSuccessTmp ){

#---- Write Document Id, Type, Name, PageCount and Size
print fIndexFile ("DocId=". $a2wDocumentPar->getId() .
", DocType=". $svDocType .
", DocName=". $svOutputFilePath . $a2wDocumentPar->getOutputFilename() .
", PageCount=". $a2wDocumentPar->getPageCount() .
", Size=". $a2wDocumentPar->getSize() . "\n");

#---- Write Indexes
for(my $IndexPtrTmp = 0; $IndexPtrTmp < $docIndexCountTmp; $IndexPtrTmp++){
print fIndexFile ("@IndexList[$IndexPtrTmp]\n");
}
print fIndexFile ("\n");
close( fIndexFile );
}
```

Listing***autosplit.pm - sub finalizeDoc() - Index-Liste für das nächste Dokument leeren***

```
#---- Reset Current Index List
@IndexList = ();
```

Das Script Beispiel ausführen

Mit den folgenden Kommandos in der Demo Stapeldatei starten Sie AFP2web mit der Scripting Facility:

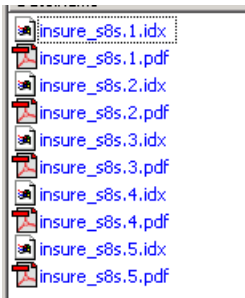
Listing
Demo Stapeldatei

```
On Windows:
afp2web.exe -q -c -doc_col d -sp: autospl i t. pm samples\i nsure. afp

On Unix:
./afp2web -q -c -doc_col d -sp: autospl i t. pm samples/i nsure. afp
```

Ergebnisse von autosplit.pm

Das Skript-Beispiel erzeugt PDF Dateien für jedes Dokument und eine zugehörige Textdatei mit Indexdaten:



Ein PDF Dokument beginnt mit einer Startseite wie folgt:



Die Textdatei mit Indexdaten hat das folgende Format:

Listing***autosplit.pm - Textdatei mit Indexdaten***

```
DocId=1, DocType=PDF, DocName=./pdf/i nsure_s8s. 1. pdf, PageCount=3,  
Si ze=126321  
PageGroup=324-1443255-1100000001  
Insured=Geoffrey R Stephens  
Pol i cy=324-1443255-11  
Page=00000001  
Contents=Di sabi l i ty Income Pol i cy  
Page=00000002  
Contents=Pol i cy Schedul e  
Contents=Pol i cy Schedul e  
Contents=Tabl e of Benefi ts  
Contents=Addi ti onal Benefi ts  
Contents=Premi um Summary  
Page=00000003  
Contents=Defi ni ti ons
```

In dieser Beschreibung gehen wir nicht näher auf die Struktur der AFP Elemente ein. Wenn Sie Ihre AFP Daten untersuchen möchten, so können Sie mit der AFP2web Kommandozeilenoption -ll eine Log-Datei erzeugen.

Das folgende Beispiel zeigt einen Ausschnitt aus einer Log-Datei, die mit der Option -ll:ALL erstellt wurde. Die Einträge für Indexdaten sind an den Beginn eines Dokuments zu finden:

```

> samples/insure.afp
}3 A8 A7 BDI Begin Document Index
    Fully Qualified Name Triplet
    Fully Qualified Name used (FQNTtype): 0x01
    This GID replaces the first parameter in the structured field that contain
    Fully Qualified Name Triplet
    Fully Qualified Name used (FQNTtype): 0x83
    The triplet contains a GID reference to a Begin Document structured field
    Fully Qualified Name Triplet
    Fully Qualified Name used (FQNTtype): 0x0a
    The triplet contains a GID reference to a Begin Resource Group structured
}3 B2 A7 IEL Index Element
    Object Byte Extent Triplet
    Byte Extent of Indexed object: 0x2ca1
    Direct Byte Offset Triplet
    Offset of the indexed object in bytes: 0x94
    Object Structured Field Extent Triplet
    Extent, in structured fields, to the indexed object: 0x62
    Object Structured Field Offset Triplet
    Offset, in structured fields, to the indexed object: 0x02
    Medium Map Page Number Triplet
    Sequence Number of The Indexed Page: 0x01
    Fully Qualified Name Triplet
    Fully Qualified Name used (FQNTtype): 0x0d
    The triplet contains a GID reference to a Begin Named Page Group structur
}3 A0 90 TLE Tag Logical Element
    Sequence Number: 00000000
    Level Number: 00000000
    Fully Qualified Name Triplet
    Fully Qualified Name used (FQNTtype): 0x0b
    The triplet contains the GID of a document attribute: Insured
    Attribute Value Triplet
    Attribute Value: Geoffrey R Stephens
}3 A0 90 TLE Tag Logical Element
    Sequence Number: 00000000
    Level Number: 00000000
    Fully Qualified Name Triplet
    Fully Qualified Name used (FQNTtype): 0x0b
    The triplet contains the GID of a document attribute: Policy
    Attribute Value Triplet
    Attribute Value: 324-1443255-11
}3 B2 A7 IEL Index Element

```

dumpPageTextObjs.pm: Text und Schriftinformationen ausgeben

Das Skriptbeispiel `dumpPageTextObjs.pm` zeigt wie man auf alle Textobjekte einer Seite zugreift und den Text in eine Datei ausgibt. Bei jeder Änderung der Schrifteinstellungen werden diese Informationen vor dem Text ausgegeben.

dumpPageTextObjs.pm verstehen

In dieser Beschreibung konzentrieren wir uns auf die Subroutinen:

<i>Subroutine</i>	<i>Beschreibung</i>
<code>sub initialize</code>	Initialisiert und erzeugt die Ausgabedatei.
<code>sub initializeDoc()</code>	Sichert das aktuelle Dokument-Objekt
<code>sub initializePage()</code>	Sichert das aktuelle Dokumentseiten-Objekte
<code>sub afp2web()</code>	AFP2web Haupteinstieg für die Dokumentseite. Durch die Wahl des Rückgabewerts weist diese Routine AFP2web an, die Seite als Beginn eines neuen Dokumentes zu nehmen oder dem aktuellen Dokument als weitere Seite hinzufügen, die aktuelle Seite zu ignorieren, oder den AFP2web Prozess zu beenden. Unser Skript-Beispiel schreibt den Text der Seite in die Ausgabedatei . Bei einer Änderung in der Schrift, werden auch die neuen Schrifteinstellungen vor dem Text ausgegeben.
<code>sub finalizeDoc()</code>	Schreibt die Indexdaten des aktuellen Dokuments in eine Ausgabedatei.

Die Subroutine `initialize()` enthält Instruktionen zur Erzeugung der Ausgabedatei:

Listing
dumpPageTextObjs.pm - initializeDoc()

```

sub initialize(){#---- Get Parameter of initialize( Par: a2w::Config,
a2w::Kernel )
( $a2wConfigPar, $a2wKernelPar ) = @_;

my $svScriptProcTmp = $a2wConfigPar->getAttribute("ScriptProcedure");
my $svScriptArgsTmp = $a2wConfigPar->getScriptArgs();
$svIndexPath = $a2wConfigPar->getIndexPath();
$svOutputFilePath = $a2wConfigPar->getOutputFilePath();
$svSpoolFilename = $a2wKernelPar->getSpoolFilename();

#---- Open Dump file
my ($svSpoolFilePathTmp, $svDumpFilenameTmp) = ($svSpoolFilename =~
/^(?:(?:[\\\/])?)(.*)$/s);
$svDumpFilenameTmp = $svOutputFilePath . $svDumpFilenameTmp . ".txt";
open( fDumpFile, ">$svDumpFilenameTmp" );
print "Running $svScriptProcTmp: Dumping to $svDumpFilenameTmp...\n";

return 0; }

```

Die Subroutine `initializeDoc()` wird aufgerufen, wenn AFP2web den Beginn eines neuen Dokuments signalisiert. Wir sichern das aktuelle Dokument-Objekt, das als Argument an diese Routine übergeben wird, in der Perl Variable `@_`.

Listing
dumpPageTextObjs.pm - initializeDoc()

```

sub initializeDoc(){
#---- Get Parameter of initializeDoc( Par: a2w::Document )
($a2wDocumentPar) = @_;

return 0; }

```

Genauso wichtig ist es, in der Routine `initializePage()` das aktuelle Dokumentseiten-Objekt zu sichern, um später auf deren Methoden zugreifen zu können:

Listing
dumpPageTextObjs.pm - initializePage()

```
sub initializePage(){#---- Get Parameter of initializePage( Par: a2w: Page
)
($a2wPagePar) = @_;

return 0; }
```

Die zentrale Verarbeitungslogik ist in der Routine `afp2web()` formuliert. Die erste Aktion ist das Setzen eines Defaults für den Rückgabewert. Auch wird die Seitenzählung aktualisiert und in die Ausgabedatei ausgegeben.

Listing
dumpPageTextObjs.pm - afp2web() - initializing the return code

```
sub afp2web(){
if ( $bLog == $TRUE ){
print "afp2web(): PageId " . $a2wPagePar->getParseId() . "\n";
}

$APPEND= 0; # append page to Current Document
$SKIP= 1; # skip page
$NEWDOC= 2; # new document

$svRetTmp = $APPEND; # default: append page

#---- Get Page Id
my $svPageIdTmp = $a2wPagePar->getParseId();

#---- Dump
printf fDumpFile ("===== ");
printf fDumpFile ("Page %06d", $svPageIdTmp);
printf fDumpFile (" =====\n");

.... additional lines not shown here....

return $svRetTmp;
}
```

Die Routine `afp2web()` wird für jede Dokumentenseite aufgerufen. In einer Schleife greifen wir auf alle Textobjekte der Seite zu und schreiben deren Text-Werte in die Ausgabedatei.

Listing
dumpPageTextObjs.pm - afp2web() - Text ausgeben

```
#---- Fetch first Text Object
my $a2wTextTmp = $a2wPagePar->getFirstText();

#---- Loop thru all the Texts
while ( $a2wTextTmp != 0 ){

.... additional lines... for dumping font information .(see next listing)....

#---- Build String to be dumped with details
printf fDumpFile (" @" . $a2wTextTmp->getXPos() . " , " . $a2wTextTmp->getYPos . " )>" . $a2wTextTmp->getText() . "<\n");

#---- Get the next Text Object
$a2wTextTmp = $a2wPagePar->getNextText();
} # end-while
```

Innerhalb der Schleife für Textobjekte untersuchen wir die Schrifteinstellungen. Ist eine Änderung festzustellen, so schreiben wir Informationen über die Schrifteinstellungen in die Ausgabedatei. Hierzu erweitern wir die Schleife wie folgt:

Listing***dumpPageTextObjs.pm - afp2web() - Schrifteinstellungen ausgeben***

```

#---- Write font entry with font details
$svTmp = $a2wTextTmp->getMappedFontLocalId();

if ( $svTmp != $svFontLocalIdTmp ){ # Compare Font Local Id against Current one

#---- Save Font Local Id as Current
$svFontLocalIdTmp = $svTmp;

#---- Fetch Font Object
$a2wFontTmp = $a2wTextTmp->getFont();

#---- Build String to be dumped
$svFontEntryTmp = "==">;

#---- Fetch Coded Font name
$svTmp = $a2wFontTmp->getCodedFontName();
if ( $svTmp ne "" ){
$svFontEntryTmp .= "CF=$svTmp, ";
}

#---- Fetch Character Set name
$svTmp = $a2wFontTmp->getCharacterSetName();
if ( $svTmp ne "" ){
$svFontEntryTmp .= "CS=$svTmp, ";
}

.... additional font attributes ....

#---- Dump Font Entry to File
printf fDumpFile (" $svFontEntryTmp\n");
}

```

Das Skript Beispiel ausführen

Mit den folgenden Kommandos in der Demo Stapeldatei starten Sie AFP2web mit der Scripting Facility:


Listing
Demo Stapeldatei

```
On Windows: afp2web.exe -q -c -doc_col d -sp: dumpPageTextObjs.pm samp-
les\insure.afp

#On Unix: ./afp2web -q -c -doc_col d -sp: dumpPageTextObjs.pm samp-
les/insure.afp#
```

Ergebnisse von dumpPageTextObjs.pm

Sie bekommen ein PDF Dokument mit einer Startseite wie folgt:



Insurance
Specialists

Disability Income Policy

The Preferred Professional

Insurance Specialists Company

100 Main Street
Anycity, Anystate 99999-9999

Insurance Specialists Company will pay the benefits provided in this Policy for loss due to Injury or Sickness.

We have issued this Policy to You in consideration of the payment of the premium and the statements made in Your application. Your application is part of Your Policy.

Insured

Geoffrey R Stephens

Policy Number

324-1443255-11

03-25-53

Date of Issue

Die vom Skript erstellte Ausgabedatei ist im Unterverzeichnis /pdf. Das folgende Beispiel zeigt einen Auszug von dem Inhalt dieser Datei:

Listing**dumpPageTextObjs.pm - Auszug aus dem Inhalt der Skript-Ausgabe**

```

===== Page 000001 =====
==>CS=C1H400B0, TF=HELVETICA LATIN1, W=B, S=12.00, CP=T1GI 0361
@(8496,862)>Insurance<
@(8496,1120)>Specialists<
==>CS=C1H400J0, TF=HELVETICA LATIN1, W=B, S=20.00, CP=T1GI 0361
@(792,2308)>Disability Income Policy<
==>CS=C1H200D0, TF=HELVETICA LATIN1, W=R, S=14.00, CP=T1GI 0361
@(792,2638)>The Preferred Professional<
==>CS=C1H40000, TF=HELVETICA LATIN1, W=B, S=10.00, CP=T1GI 0361
@(7330,2209)>Insurance Specialists Company<
==>CS=C1H20090, TF=HELVETICA LATIN1, W=R, S=9.00, CP=T1GI 0361
@(7330,2418)>100 Main Street<
@(7330,2627)>Any city, Anystate 99999-9999<
==>CS=C1H20000, TF=HELVETICA LATIN1, W=R, S=10.00, CP=T1GI 0361
@(792,4104)>Insurance Specialists Company will pay the benefits provided
in this Policy for loss due to Injury or Sickness.<
@(792,4471)>We have issued this Policy to You in consideration of the
payment of the premium and the statements made in<
@(792,4681)>Your application. Your application is part of Your Policy.<

```

sortPageTextObjs.pm: Textobjekte nach der Print-Reihenfolge sortieren

In einem Spool mit AFP Druckdaten werden die Textobjekte nicht immer in der Reihenfolge wiedergegeben, wie sie ausgedruckt werden. Dies erschwert das Lesen von AFP Datenausgängen.

Das Skript-Beispiel *sortPageTextObjs.pm* zeigt, wie Textobjekte vor der Ausgabe in eine Textdatei zunächst in der Reihenfolge ihrer X/Y Koordinatenpositionen auf der Druckseite sortiert werden.

sortPageTextObjs.pm verstehen

In dieser Beschreibung konzentrieren wir uns auf die folgenden Subroutinen:

<i>Subroutine</i>	<i>Beschreibung</i>
sub afp2web()	Haupteinstieg für AFP2web. Durch die Wahl des Rückgabewerts weist diese Routine AFP2web an, die Seite als Beginn eines neuen Dokumentes zu nehmen oder dem aktuellen Dokument als weitere Seite hinzufügen, die aktuelle Seite zu ignorieren, oder den AFP2web Prozess zu beenden. Unser Skript sammelt die Textobjekte der Dokumentseite und ruft die Routine buildLines() um diese zu sortieren und in eine Textdatei auszugeben.
sub buildLines()	Sortiert Text nach der Y-Koordinate (Zeilenummer) und dann nach der X-Koordinate (Position innerhalb der Zeile).
sub complex_arrays()	Legt die Schlüssel für den Sortieralgorithmus fest.

Die Subroutine *afp2web()* ist der Haupteinstieg für AFP2web. Sie wird für jede Dokumentenseite aufgerufen.

Die erste Aktion ist die Initialisierung des Rückgabewerts. Eine weitere Aktion ist die Ausgabe der aktuellen Seitennummer.

Im Folgenden zeigen wir wie *afp2web()* alle Textobjekte der aktuellen Seite sammelt und die Subroutine *buildLines()* zur Sortierung aufruft:

Listing***sortPageTextObjs.pm - afp2web() - Textobjekte sammeln und verarbeiten***

```

#---- Add the Page Text Objects to an unsorted Object List
@unsortedObjList = ();
my $PtrTmp = @unsortedObjList;
#---- Define temp variables
my $svTextTmp = "";
my $svTextXPosTmp = 0;
my $svTextYPosTmp = 0;

#---- Fetch first Text Object
my $a2wTextTmp = $a2wPagePar->getFirstText();

#---- Loop thru all the Text Objects
while ( $a2wTextTmp != 0 ){
    $svTextTmp = $a2wTextTmp->getText();
    $svTextXPosTmp = $a2wTextTmp->getXPos();
    $svTextYPosTmp = $a2wTextTmp->getYPos();
    if ( $bLog == $TRUE ){
        print " @(" . $svTextXPosTmp . "," . $svTextYPosTmp . ">" . $svTextTmp .
        "<\n";
    }
    #---- Add Object to unsorted Object List
    @unsortedObjList[$PtrTmp++] = ({XPOS => $svTextXPosTmp,
    YPOS => $svTextYPosTmp,
    TEXT => $svTextTmp
    });
    #---- Get the next Text Object
    $a2wTextTmp = $a2wPagePar->getNextText();
} # end-while

#---- Build/Dump Lines
buildLines();

```

Die Subroutine buildLines() sortiert den Text und schreibt diese dann in der Sortierreihenfolge in die Ausgabedatei. Die Y-Koordinate legt die vertikale Position (Zeilennummer) fest. Jede neue Zeilennummer wird ebenfalls mit ausgegeben.

Listing
sortPageTextObjs.pm - buildLines()

```
sub buildLines(){

#---- Define temp variables
my $LineTmp = "";
my $LineCountTmp = 0;
my $currYPosTmp = 0;
my @sortedObjList = ();

#---- Sort Array: first Key is YPOS, second Key is XPOS
@sortedObjList = sort complex_arrays @unsortedObjList;

#---- Build Lines based on sorted Array
foreach (@sortedObjList){

$YPosTmp = $_->{YPOS};
$TextTmp = $_->{TEXT};

if ( $currYPosTmp != $YPosTmp && $currYPosTmp != 0 ){

#---- Add line
printf fDumpFile ("%06d:", $LineCountTmp);
print fDumpFile (" $LineTmp\n");

$LineCountTmp++;
$LineTmp = $TextTmp;
}
else{ # append Text to current line
$LineTmp = $LineTmp . $TextTmp;
}
$currYPosTmp = $YPosTmp;
}
}
```

Die Subroutine `complex_arrays()` liefert die Sortierschlüssel für das Sort Kommando. In unserm Falle wird zuerst nach der Y-Koordinate und dann nach der X-Koordinate sortiert.

Listing***sortPageTextObjs.pm - complex_arrays()***

```
#-----  
# Sorting Algorithm  
#  
#1st Key is YPOS, 2nd Key is XPOS  
#-----  
sub complex_arrays{  
  $a->{YPOS} <=> $b->{YPOS} ||  
  $a->{XPOS} <=> $b->{XPOS};  
}
```

Das Skript Beispiel ausführen

Sie verwenden die folgenden Kommandos in der Demo Stapeldatei um AFP2web mit der Scripting Facility zu starten:

Listing***Demo Stapeldatei***

```
On Windows:  
afp2web.exe -q -c -doc_col d -sp: sortPageTextObjs.pm samples\insure.afp  
  
On Unix:  
./afp2web -q -c -doc_col d -sp: sortPageTextObjs.pm samples/insure.afp
```

Ergebnisse von sortPageTextObjs.pm

Das resultierende PDF Dokument beginnt mit einer Startseite wie folgt:



Insurance
Specialists

Disability Income Policy

The Preferred Professional

Insurance Specialists Company
 100 Main Street
 Anycity, Anystate 99999-9999

Insurance Specialists Company will pay the benefits provided in this Policy for loss due to Injury or Sickness.
 We have issued this Policy to You in consideration of the payment of the premium and the statements made in Your application. Your application is part of Your Policy.

Insured Geoffrey R Stephens

Policy Number 324-1443255-11 03-25-53 **Date of Issue**

Die vom Skript erstellte Ausgabedatei ist im Unterverzeichnis /pdf. Das folgende Beispiel zeigt einen Auszug von dem Inhalt dieser Datei. Jede Textzeile beginnt mit der Zeilennummer. (Hinweis: In unserer Abbildung wurden aus Platzgründen zusätzliche Zeilenumbrüche eingefügt.)

Listing***sortPageTextObjs.pm - Auszug aus der Ausgabedatei***

```
===== Page 000001 = =====
000000: Demo Version
000001: Specialists
000002: Insurance Specialists Company
000003: Disability Income Policy
000004: 100 Main Street
000005: Anycity, Anystate 99999-9999
000006: Demo Version
000007: Insurance Specialists Company will pay the benefits provided in
this Policy for loss due to Injury or Sickness.
000008: We have issued this Policy to You in consideration of the payment
of the premium and the statements made in
000009: Your application. Your application is part of Your Policy.
000010: Demo VersionGeoffrey R Stephens
000011: Policy Number324-1443255-1103-25-53Demo Version
000012: Demo Version
000013: As long as the premium is paid on time, We cannot change Your
Policy or its premium rate until the first premium
000014: due date after Your 65th birthday.
000015: RENEWAL OPTIONS AFTER YOU REACH AGE 65. SUBJECT TO CHANGE IN PRE-
MIUM RATES. Demo Version
000016: 65 to age 72, You may continue Your Policy for a Total Disability
benefit with a limited benefit period while You

..... weitere Zeilen hier nicht gezeigt....
```

dumpMediumMaps.pm: Liste der Medium Maps ausgeben

Dieses Skript-Beispiel erstellt eine Textdatei mit einer Liste aller Medium Maps zu einer Dokumentenseite.

dumpMediumMaps.pm verstehen

In dieser Beschreibung konzentrieren wir uns auf die Subroutinen:

<i>Subroutine</i>	<i>Beschreibung</i>
sub afp2web()	Haupteinstieg für AFP2web. Durch die Wahl des Rückgabewerts weist diese Routine AFP2web an, die Seite als Beginn eines neuen Dokumentes zu nehmen oder dem aktuellen Dokument als weitere Seite hinzufügen, die aktuelle Seite zu ignorieren, oder den AFP2web Prozess zu beenden. Wir zeigen wie afp2web() Informationen über Medium Maps der Seite ermittelt und ausgibt.

Die Subroutine afp2web() ist der Haupteinstieg für AFP2web und wird für jede Dokumentenseite aufgerufen.

Die Kommandos zur Ausgabe von Informationen über das aktuelle Medium Map:

Listing

dumpMediumMaps.pm - afp2web() -

```
#---- Get Page Id
my $svPageIdTmp = $a2wPagePar->getParseId();

#---- Dump
printf fDumpFile ("===== ");
printf fDumpFile ("Page %06d", $svPageIdTmp);
printf fDumpFile (" =====\n");

#---- Fetch applied medium map
my $a2wMediumMapTmp = $a2wPagePar->getAppliedMediumMap();

if ( $a2wMediumMapTmp != 0 ){

#---- Dump Formdef Name
printf fDumpFile ("Formdef: " . $a2wMediumMapTmp->getFormdefName() .
"\n");

#---- Dump Medium Map Name
printf fDumpFile ("Medium Map: " . $a2wMediumMapTmp->getName() . ", ");

#---- Dump Medium Map Width, Height, Resolution, Duplex Control and N-Up
Control
printf fDumpFile ("Width=" . $a2wMediumMapTmp->getWidth() .
", Height=" . $a2wMediumMapTmp->getHeight() .
", Resolution=" . $a2wMediumMapTmp->getResolution() .
", Duplex Control=" . $a2wMediumMapTmp->getDuplexControl() .
", N-Up Control=" . $a2wMediumMapTmp->getNupControl() .
"\n");
}
```

Das Script Beispiel ausführen

Sie verwenden die folgenden Kommandos in der Demo Stapeldatei um AFP2web mit der Scripting Facility zu starten:


Listing
Demo Stapeldatei

```
On Windows:
afp2web.exe -q -c -doc_col d -sp: dumpMediumMaps.pm samples\insure.afp

On Unix:
./afp2web -q -c -doc_col d -sp: dumpMediumMaps.pm samples/i nsure.afp
```

Ergebnisse von dumpMediumMaps.pm

Das resultierende PDF Dokument beginnt mit einer Startseite wie folgt:

 Insurance Specialists	
<div style="border-bottom: 2px solid black; margin-bottom: 10px;"></div> <div style="display: flex; justify-content: space-between;"> <div style="color: green; font-weight: bold; font-size: 1.2em;">Disability Income Policy</div> <div style="text-align: right; font-weight: bold; font-size: 0.9em;">Insurance Specialists Company</div> </div> <div style="display: flex; justify-content: space-between;"> <div style="color: red; font-weight: bold;">The Preferred Professional</div> <div style="text-align: right; font-weight: normal; font-size: 0.8em;">100 Main Street Anycity, Anystate 99999-9999</div> </div>	
<p>Insurance Specialists Company will pay the benefits provided in this Policy for loss due to Injury or Sickness.</p> <p>We have issued this Policy to You in consideration of the payment of the premium and the statements made in Your application. Your application is part of Your Policy.</p>	
<div style="display: flex; justify-content: space-between;"> <div> <p>Insured Geoffrey R Stephens</p> </div> </div>	
<div style="display: flex; justify-content: space-between;"> <div> <p>Policy Number 324-1443255-11</p> </div> <div> <p>03-25-53</p> </div> <div> <p>Date of Issue</p> </div> </div>	

Die vom Skript erstellte Ausgabedatei ist im Unterverzeichnis /pdf. Das folgende Beispiel zeigt einen Auszug von dem Inhalt dieser Datei. In unserem Beispiel wird das gleiche Medium Map für jede Dokumentseite verwendet.

Listing

dumpMediumMaps.pm - Auszug aus der Ausgabedatei

```
===== Page 000001
=====
Formdef: F1DEMO
Medium Map: OGL, Width=0, Height=0, Resolution=240, Duplex Control=1, N-Up
Control=1
===== Page 000002
=====
Formdef: F1DEMO
Medium Map: OGL, Width=0, Height=0, Resolution=240, Duplex Control=1, N-Up
Control=1
===== Page 000003
=====
Formdef: F1DEMO
Medium Map: OGL, Width=0, Height=0, Resolution=240, Duplex Control=1, N-Up
Control=1

.... weitere Zeilen hier nicht gezeigt. ....
```

dumpNOPs.pm: Inhalte von NOP Feldern ausgeben

NOP Felder werden verwendet, um einen Datenstrom mit nicht sichtbaren Dokumentinformationen anzureichern. Unterschiedliche Anwendungen haben ihre eigenen Konventionen für NOP Felder und verwenden diese beispielsweise für Indizierung, Archivierung und für die Verarbeitungssteuerung.

Möglicherweise möchten Sie die vorhandenen NOP-Informationen in Ihren AFP Spools untersuchen.

dumpNOPs.pm erstellt eine Auflistung der Werte, die in den NOP-Feldern einer Seite mitgegeben werden.

dumpNOPs.pm verstehen

In dieser Beschreibung konzentrieren wir uns auf die Subroutinen:

<i>Subroutine</i>	<i>Beschreibung</i>
sub afp2web()	Haupteinstieg für AFP2web. Durch die Wahl des Rückgabewerts weist diese Routine AFP2web an, die Seite als Beginn eines neuen Dokumentes zu nehmen oder dem aktuellen Dokument als weitere Seite hinzufügen, die aktuelle Seite zu ignorieren, oder den AFP2web Prozess zu beenden. Wir zeigen wie afp2web() alle NOP Felder der aktuellen Seite verarbeitet.

Die Subroutine afp2web() ist der Haupteinstieg für AFP2web und wird für jede Dokumentenseite aufgerufen. Gesammelt werden alle NOP-Objekte der aktuellen Seite. Ausgegeben werden deren NOP-Werte.

Listing
dumpNOPs.pm - afp2web() - ... NOP Werte ausgeben

```
#---- Get Page Id
my $svPageIdTmp = $a2wPagePar->getParseId();

#---- Dump
printf fDumpFile ("===== ");
printf fDumpFile ("Page %06d", $svPageIdTmp);
printf fDumpFile (" =====\n");

#---- Fetch 1st NOP
my $a2wNOPTmp = $a2wPagePar->getFirstNOP();

if ( $a2wNOPTmp != 0 ){
my $svNOPDataTmp = $a2wNOPTmp->getValue();

print fDumpFile ("NOP=$svNOPDataTmp\n");

#---- Fetch next NOP
$a2wNOPTmp = $a2wPagePar->getNextNOP();

#---- Loop for each and every NOP
while ( $a2wNOPTmp != 0 ){

$svNOPDataTmp = $a2wNOPTmp->getValue();

#---- Write NOP to dump file
print fDumpFile ("NOP=$svNOPDataTmp\n");

#---- Fetch next NOP
$a2wNOPTmp = $a2wPagePar->getNextNOP();
}; # end-while
}
```

Das Script Beispiel ausführen

Sie verwenden die folgenden Kommandos in der Demo Stapeldatei um AFP2web mit der Scripting Facility zu starten:


Listing
Demo Stapeldatei

```
On Windows:
afp2web.exe -q -c -doc_col d -sp: dumpNOPs. pm samples\i nsure. afp

On Unix:
./afp2web -q -c -doc_col d -sp: dumpNOPs. pm samples/i nsure. afp
```

Ergebnisse von dumpNOPs.pm

Das resultierende PDF Dokument beginnt mit einer Startseite wie folgt:

		 Insurance Specialists
<hr/> Disability Income Policy The Preferred Professional		Insurance Specialists Company 100 Main Street Anycity, Anystate 99999-9999
<hr/>		
Insurance Specialists Company will pay the benefits provided in this Policy for loss due to Injury or Sickness. We have issued this Policy to You in consideration of the payment of the premium and the statements made in Your application. Your application is part of Your Policy.		
Insured	Geoffrey R Stephens	
Policy Number	324-1443255-11	03-25-53 Date of Issue
NON-CANCELLABLE AND GUARANTEED CONTINUABLE TO AGE 65. NO CHANGE IN PREMIUM RATES.		

Die vom Skript erstellte Ausgabedatei ist im Unterverzeichnis /pdf/. Das folgende Beispiel zeigt einen Auszug von dem Inhalt dieser Datei. Ausgegeben wird jeweils die Seitennummer und alle NOP-Felder der Seite. In unserm Beispiel finden wir nur ein NOP-Objekt:

Listing
dumpNOPs.pm - Auszug aus der Ausgabedatei

```
===== Page 000001 =====
NOP=SCRIPT/VS 4.0.0: DEVICE AFP4 CHARS XON2200E
===== Page 000002 =====
===== Page 000003 =====
===== Page 000004 =====
===== Page 000005 =====
===== Page 000006 =====
===== Page 000007 =====
===== Page 000008 =====
===== Page 000009 =====
===== Page 000010 =====
===== Page 000011 =====
===== Page 000012 =====
===== Page 000013 =====
===== Page 000014 =====
===== Page 000015 =====
```

Mit der Kommandozeilenoption -ll:ALL erstellen wir eine Log-Datei. Ein Blick in der Log-Datei zeigt, dass unser Beispiel tatsächlich nur ein NOP-Objekt enthält:

The screenshot shows a PDF viewer window with a search dialog open. The search dialog, titled "Lines containing find string:", lists parameters and search results. The first result is "0x00000FE9: D3 EE EE NOP No Operation". The PDF content in the background shows a log file with various entries, including "NOP No Operation" and "BNG Begin Named Page Group 00000001".

Search Dialog Content:

```
List of Parameters: -q -c -fm -doc -cold -sp dumpNOPs.pm -ll:ALL -a:dump
0x00000FE9: D3 EE EE NOP No Operation
0x00001230: D3 EE EE NOP No Operation
PTOCA NOP: no data found
PTOCA NOP: no data found
PTOCA NOP: no data found
PTOCA NOP: no data found
PTOCA NOP: no data found
PTOCA NOP: no data found
PTOCA NOP: no data found
PTOCA NOP: no data found
PTOCA NOP: no data found
```

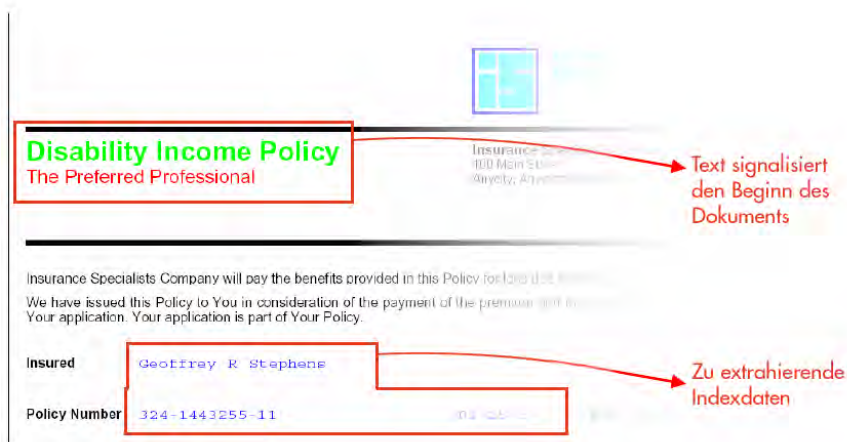
PDF Content (Log File):

```
757 Interchange Set Type: Pr
758 Interchange set identifi
759
760 Fully Qualified Name Tri
761 Fully Qualified Name use
762 This GID replaces the fi
763 Fully Qualified Name Tri
764 Fully Qualified Name use
765 The triplet contains a G
766 Fully Qualified Name Tri
767 Fully Qualified Name used (FontType): 0x98
768 The triplet contains a GID reference to a Begin Document Index structured field: SMLIN
769 0x00000FE9 : D3 EE EE NOP No Operation
770 E2 C3 D9 C9 D7 E3 61 E5 E2 40 F4 4B F0 4B F0 7A 40 C4 C5 E5 C9 C3 C5 40 C1 C6 D7 C1
771 F0 F0 C5
772 SCRIPT/VS 4.0.0: DEVICE AFP4 CHARS XON2200E
773 0x0000101D : D3 A8 AD BNG Begin Named Page Group 00000001
```

eyecatcher.pm: Text Finden und Extrahieren

eyecatcher.pm findet und extrahiert vordefinierte Zeichenfolgen im Text einer Seite. Jede Zeichenfolge hat eine vorgegebene X/Y Koordinatenposition. Dieses Skriptbeispiel erkennt diese Zeichenfolgen entweder :

- als Signal für den Beginn eines neuen Dokuments oder
- als eine Stelle zum Herauslesen von Indexdaten aus dem Dokumenteninhalt.



Mit eyecatcher.pm Koordinatenpositionen der Textobjekte ermitteln

Wir ermitteln die Koordinatenpositionen der gesuchten Texte, indem wir mit der AFP2web Scripting Facility die folgenden Kommandos in der Skript-Subroutine `afp2web()` ausführen:

Listing
eyecatcher.pm - Ausgabe der Text-Koordinaten

```
sub afp2web(){
$APPEND = 0; # append page to Current Document
$SKIP = 1; # skip page
$NEWDOC = 2; # new document
#---- Set default return value
my $svRetTmp = $APPEND; # default: append page
#---- Fetch first Text Object
my $a2wTextTmp = $a2wPagePar->getFirstText();
#---- Define temp variables
my $svTextTmp = "";
my $svTextXPosTmp = 0;
my $svTextYPosTmp = 0;
#---- Loop thru all the Text Objects
while ( $a2wTextTmp != 0 ){
$svTextTmp = $a2wTextTmp->getText();
$svTextXPosTmp = $a2wTextTmp->getXPos();
$svTextYPosTmp = $a2wTextTmp->getYPos();
print $svTextXPosTmp . "," . $svTextYPosTmp . ">" . $svTextTmp . "\n";
#---- Fetch next Text Object
$a2wTextTmp = $a2wPagePar->getNextText();
}
#---- Increment Page Id
$PageId++;

return $svRetTmp;
}
```

Mit den folgenden Befehlen in demo.bat starten wir AFP2web mit der Scripting Facility:

Listing
Demo.bat

```
afp2web.exe -q -c -doc_cold -sp:eyecatcher.pm samples\insure.afp > dump-
file.txt
```

Das Skript produziert die folgende Ausgabe, aus der wir die Koordinatenpositionen unserer gesuchten Texte ermitteln:

Listing eyecatcher.pm -
<pre> 8496, 862>Insurance 8496, 1120>Specialists 792, 2308>Disability Income Policy 792, 2638>The Preferred Professional 7330, 2209>Insurance Specialists Company 7330, 2418>100 Main Street 7330, 2627>Anycity, Anystate 99999-9999 792, 4104>Insurance Specialists Company will pay the benefits provided in this Policy for loss due to Injury or Sickness. 792, 4471>We have issued this Policy to You in consideration of the payment of the premium and the statements made in 792, 4681>Your application. Your application is part of Your Policy. 792, 5353>Insured 2448, 5353>Geoffrey R Stephens 792, 6097>Policy Number 2448, 6097>324-1443255-11 weitere Zeilen hier nicht gezeigt. </pre>

Mit eyecatcher.pm Dokumente trennen und Indexdaten extrahieren

In dieser Beschreibung konzentrieren wir uns auf die Subroutinen:

Subroutine	Beschreibung
sub initializeDoc()	Sichert das aktuelle Dokumenten-Objekt Setzt die Seitenzählung zurück auf Null.
sub afp2web()	Durch die Wahl des Rückgabewerts weist diese Routine AFP2web an, die Seite als Beginn eines neuen Dokumentes zu nehmen oder dem aktuellen Dokument als weitere Seite hinzu- fügen, die aktuelle Seite zu ignorieren, oder den AFP2web Prozess zu beenden. An dieser Stelle wird der Beginn einer neuen Seite festgestellt. Es wird die Subroutine addPageEyeCatcherIndexes() für die erste Seite des Dokuments aufgerufen.
sub finalizeDoc()	Schreibt Indexdaten zum Dokument in die Ausgabedatei.

Subroutine	Beschreibung
sub addPageEyeCatcherIndexes()	Routine zur Extrahierung von Textwerten aus der aktuellen Seite und zur Erstellung von Indexdaten.

Die Subroutine initializeDoc() wird aufgerufen, wenn AFP2web den Beginn eines neuen Dokuments signalisiert. Wir speichern das Dokument-Objekt, das als Argument übergeben wird, in der Perl Variable @_ . Wir setzen die Seitenzählung zurück auf Null.

Listing
eyecatcher.pm - initializeDoc()

```
sub initializeDoc(){  
  
#---- Get Parameter of initializeDoc( Par: a2w::Document )  
($a2wDocumentPar) = @_  
  
return 0;  
}
```

In initializePage() sichern wir auch eine Referenz auf die aktuelle Dokumentenseite, um später auf die Methoden dieses Objekts zugreifen zu können.

Listing
eyecatcher.pm - initializePage()

```
sub initializePage(){  
  
#---- Get Parameter of initializePage( Par: a2w::Page )  
($a2wPagePar) = @_  
  
}
```

Mit der Routine afp2web() wird jede Dokumentenseite verarbeitet. Zur ersten Aktion gehört die Initialisierung des Rückgabewerts an AFP2web.

Listing***eyecatcher.pm - afp2web() - Initialisierung des Rückgabewerts***

```

#-----
# Main entry method
# Return values:
# < 0: error
# 0: append page to Current Document
# 1: skip page
# 2: first page / new document
#-----
sub afp2web(){

$APPEND= 0; # append page to Current Document
$SKIP= 1; # skip page
$NEWDOC= 2; # new document

$svRetTmp = $APPEND; # default: append page

.... weitere Zeilen mit Verarbeitungslogik hier nicht gezeigt. ....

return $svRetTmp;
}

```

Die Verarbeitungslogik in der Routine `afp2web()` sorgt für die Trennung des AFP Spools in einzelne Dokumente. Die Regel lautet: Die Startseite eines Dokuments enthält die Zeichenfolge "Disability Income Policy" an einer fixen Koordinatenposition auf der Seite. Wird diese Zeichenfolge gefunden, so setzen wir den entsprechenden Rückgabewert, um AFP2web zu veranlassen, die Verarbeitung für das vorige Dokument abzuschließen und mit dem neuen Dokument zu beginnen.

Listing
eyecatcher.pm afp2web() - Eyecatcher für die Dokumenttrennung

```
.....
#---- Fetch first Text Object
my $a2wTextTmp = $a2wPagePar->getFirstText();

#---- Loop thru all the Text Objects
while ( $a2wTextTmp != 0 ){

#---- If true ==> new doc
if ( $a2wTextTmp->getXPos() >= 782 &&
$a2wTextTmp->getXPos() <= 802 &&
$a2wTextTmp->getYPos() >= 2298 &&
$a2wTextTmp->getYPos() <= 2318 &&
$a2wTextTmp->getText() eq "Di sability Income Policy" ){

#---- Reset Page Id
$PageId = 0;

$svRetTmp = $NEWDOC;
last; # leave while loop
}

#---- Fetch next Text Object
$a2wTextTmp = $a2wPagePar->getNextText();
}

#---- Increment Page Id
$PageId++;
```

Wenn AFP2web die Verarbeitung für ein Dokument abschließt, startet es die Subroutine `finalizeDoc()`. In unserem Beispiel werden in dieser Routine die Indexdaten gelesen und ausgegeben. Diese einfache Routine ruft die Subroutine `addFirstPageEyecatcherIndexes()` auf. Sie sehen, dass Sie selbst am Ende eines Dokuments mit einfachen Methoden auf die Dokumentinhalte zugreifen können. Um die Indexdaten aus der Startseite herauszulesen, greifen Sie in diesem Falle mit der Methode `$a2wDocumentPar->getFirstPage()` auf die erste Seite zu und untersuchen dann deren Inhalt.

Listing***eyecatcher.pm - addFirstPageEyecatcherIndexes() aufgerufen von finalizeDoc()***

```
sub addFirstPageEyecatcherIndexes(){

#---- Fetch the first Page of Document
my $a2wFirstPageTmp = $a2wDocumentPar->getFirstPage();

#---- If a Page exists...
if ( $a2wFirstPageTmp != 0 ){

#---- Define the list of Eyecatcher positions we should look for
my @arrEyeCatcherIndexListTmp = ( ( 2448, 5353 ), # Insured
@(2448, 5353)>Geoffrey R Stephens<
( 2448, 6097 ), # Policy Number @(2448, 6097)>324-1443255-11<
( 7114, 6097 ) ); # Date of Issue @(7114, 6097)>03-25-53<

#---- Define temp variables
my $arrECILLenTmp = @arrEyeCatcherIndexListTmp / 2;

.... additional lines not shown here...
```

Listing

eyecatcher.pm - addFirstPageEyecatcherIndexes() aufgerufen von finalizeDoc()

```
#---- Get the Index list count
my $IndexPtrTmp = @IndexList;

#---- Fetch first Text Object
my $a2wTextTmp = $a2wFirstPageTmp->getFirstText();

#---- Loop thru all the Text Objects
while ( $a2wTextTmp != 0 ){

    $svTextTmp = $a2wTextTmp->getText();
    $svTextXPosTmp = $a2wTextTmp->getXPos();
    $svTextYPosTmp = $a2wTextTmp->getYPos();

    #---- Search all the defined Eyecatchers
    for (my $i = 0; $i < $arrECILLenTmp; $i++ ){
        $svArrPosTmp = $i * 2;
        $svIdxTextXTmp = $arrEyeCatcherIndexListTmp[ $svArrPosTmp ];
        $svIdxTextYTmp = $arrEyeCatcherIndexListTmp[ $svArrPosTmp + 1 ];

        #---- If true ==> we found one of the defined Eyecatchers
        if ( $svTextXPosTmp >= ( $svIdxTextXTmp - 10 ) &&
            $svTextXPosTmp <= ( $svIdxTextXTmp + 10 ) &&
            $svTextYPosTmp >= ( $svIdxTextYTmp - 10 ) &&
            $svTextYPosTmp <= ( $svIdxTextYTmp + 10 ) ){

            if ( $i == 0 ){
                $svIndexValueTmp = "Insured=" . $svTextTmp;
            } elsif ( $i == 1 ){
                $svIndexValueTmp = "Policy Number=" . $svTextTmp;
            } elsif ( $i == 2 ){
                $svIndexValueTmp = "Date of Issue=" . $svTextTmp;
            }

            #---- Add Eyecatcher to Index List
            @IndexList[$IndexPtrTmp++] = $svIndexValueTmp;

            if ( $bLog == $TRUE ){
                print "$svIndexValueTmp\n";
            }
        }

        #---- Fetch next Text Object
        $a2wTextTmp = $a2wFirstPageTmp->getNextText();
    }
}
```

Das Script Beispiel ausführen

Sie verwenden die folgenden Kommandos in der Demo Stapeldatei um AFP2web mit der Scripting Facility zu starten:

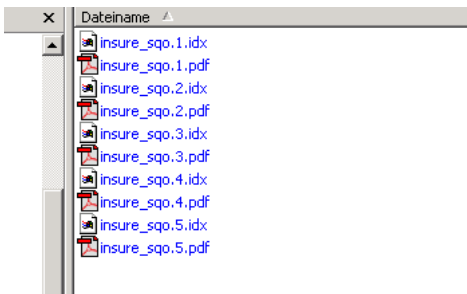
Listing **Demo Stapeldatei**

```
On Windows:
afp2web.exe -q -c -doc_col d -sp: eyecatcher.pm samples\insure.afp

On Unix:
./afp2web -q -c -doc_col d -sp: eyecatcher.pm samples/insure.afp
```

Ergebnisse von eyecatcher.pm

Sie finden das Konvertierungsergebnis und die Indexdateien standardmäßig im Unterverzeichnis /pdf:



Die Inhalte der Indexdatei anhand eines Beispiels:

Listing **eyecatcher.pm - Inhalte der Indexdatei**

```
DocId=1, DocType=PDF, DocName=./pdf/insure_sqo.1.pdf, PageCount=3,
Size=114669
Insured=Geoffrey R Stephens
Policy Number=324-1443255-11
Date of Issue=03-25-53
```

Ein PDF Dokument beginnt mit einer Startseite wie folgt:



Disability Income Policy

The Preferred Professional

Insurance Specialists Company
100 Main Street
Anycity, Anystate 99999-9999

Insurance Specialists Company will pay the benefits provided in this Policy for loss due to Injury or Sickness.
We have issued this Policy to You in consideration of the payment of the premium and the statements made in Your application. Your application is part of Your Policy.

Insured Geoffrey R Stephens

Policy Number 324-1443255-11 03-25-53 Date of Issue

NON-CANCELLABLE AND GUARANTEED CONTINUABLE TO AGE 65. NO CHANGE IN PREMIUM RATES.

addBookmarks.pm: PDF Bookmarks (Lesezeichen) erzeugen

addBookmarks.pm erstellt Lesezeichen in einem PDF Dokument.

addBookmarks.pm verstehen

In dieser Beschreibung konzentrieren wir uns auf die Subroutinen:

<i>Subroutine</i>	<i>Beschreibung</i>
sub initializeDoc()	Sichert eine Referenz auf das Dokument-Objekt Initialisiert die Seitenzählung. Erstellt PDF Bookmarks für das Dokument.
sub afp2web()	Haupteinstieg für AFP2web. Durch die Wahl des Rückgabewerts weist diese Routine AFP2web an, die Seite als Beginn eines neuen Dokumentes zu nehmen oder dem aktuellen Dokument als weitere Seite hinzufügen, die aktuelle Seite zu ignorieren, oder den AFP2web Prozess zu beenden. Erstellt PDF Bookmarks für die Seite.
sub initialize()	Aktiviert AutoSplit und die PDFBookmark Funktion. Wir zeigen diese Routine nicht in dieser Beschreibung.

Die Subroutine initializeDoc() wird aufgerufen, wenn AFP2web den Beginn eines Dokuments signalisiert. Das Dokument-Objekt, das als Argument an die Subroutine übergeben wird, sichern wir in der Perl Variable @_ . Die Seitenzählung wird auf Null zurückgesetzt. Die folgenden Zeilen zeigen wie PDF Bookmarks für das Dokument erzeugt werden:

Listing***addBookmarks.pm - initializeDoc()***

```
#---- Add document level bookmarks ----#
if ($a2wDocumentPar->getId() == 1){
#---- Add creator
$a2wDocumentPar->addBookmark( "Creator", "Maas Hi gh Tech Software GmbH" );

#---- Add creation date & time
my $svLocal TimeTmp = local time;
$a2wDocumentPar->addBookmark( "Created On", $svLocal TimeTmp );

#---- Add spool filename
$a2wDocumentPar->addBookmark( "Spool ", $svSpool Fi l e n a m e );
return 0;
}
```

In der Subroutine `initializePage()` sichern wir eine Referenz auf die Objekt der Dokumentenseite, um später dessen Methoden nutzen zu können:

Listing***addBookmarks.pm - initializePage()***

```
sub initializePage(){

#---- Get Parameter of initializePage( Par: a2w::Page )
($a2wPagePar) = @_;
```

Die Subroutine `afp2web()` verarbeitet jede Dokumentenseite. Der Rückgabewert an `AFP2web` wird hier festgelegt. PDF Bookmarks werden der Seite hinzugefügt.

Listing***addBookmarks.pm - afp2web() - PDF Bookmarks hinzufügen***

```

#-----
# Main entry method
# Return values:
# < 0: error
# 0: append page to Current Document
# 1: skip page
# 2: first page / new document
#-----
sub afp2web(){

if ( $bLog == $TRUE ){
print "afp2web(): PageId " . $a2wPagePar->getParseId() . "\n";
}

$APPEND= 0; # append page to Current Document
$SKIP= 1; # skip page
$NEWDOC= 2; # new document

#---- Set default return value
my $svRetTmp = $APPEND; # default: append page

#---- Add page level bookmarks ----#
if ($a2wDocumentPar->getId() == 1){

#---- Add page number (in spool)
$a2wPagePar->addBookmark( "Page", $a2wPagePar->getParseId() );
}
return $svRetTmp;

}

```

Das Script Beispiel ausführen

Sie verwenden die folgenden Kommandos in der Demo Stapeldatei um AFP2web mit der Scripting Facility zu starten:

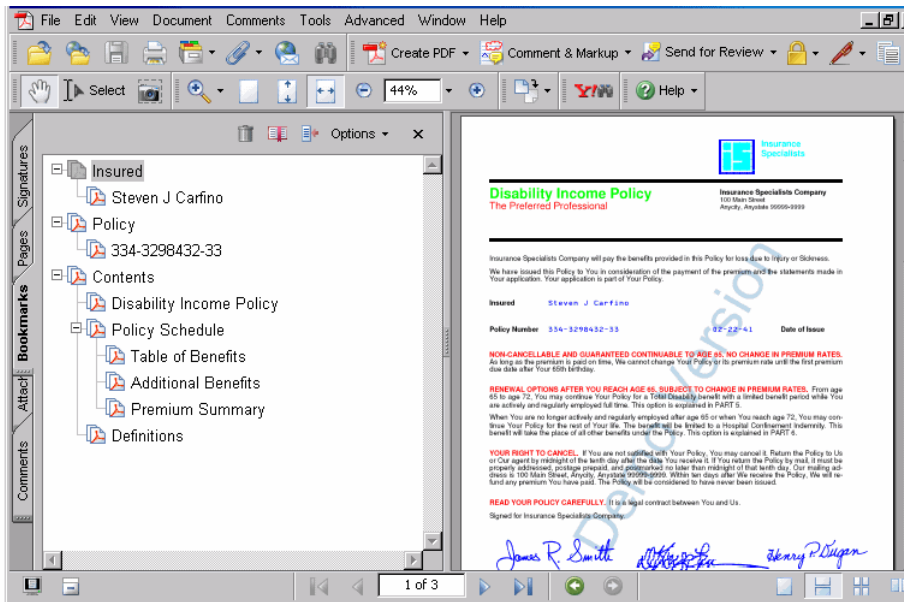
Listing Demo Stapeldatei

```
On Windows:
afp2web.exe -q -c -doc_col d -sp: addBookmarks.pm samples\insure.afp

On Unix:
./afp2web -q -c -doc_col d -sp: addBookmarks.pm samples\insure.afp
```

Ergebnisse von addBookmarks.pm

Sie finden das Konvertierungsergebnis standardmäßig im Unterverzeichnis /pdf. Ein PDF Dokument im Acrobat Reader zeigt die PDF Bookmarks wie folgt:



addWatermark.pm: Eine Hintergrundgrafik auf der Seite einfügen

Dieses Skript zeigt, wie man Text als Wasserzeichen auf einer Seite platziert.

addWatermark.pm verstehen

In dieser Beschreibung konzentrieren wir uns auf die Subroutinen:

<i>Subroutine</i>	<i>Beschreibung</i>
sub afp2web()	Haupteinstieg für AFP2web. In afp2web() wird das Wasserzeichen hinzugefügt.

Die folgenden Anweisungen positionieren Text als Wasserzeichen auf der Seite:

Listing *addWatermark.pm - sub afp2web()*

```
#---- Add Watermark ----#
#---- Create Helvetica 60 points Font (Watermark font)
$a2wFontTmp = new a2w::Font();
$a2wFontTmp->setName("Helvetica");
$a2wFontTmp->setHeight(60);
$a2wFontTmp->setBold($TRUE);

#---- Create Text Object ----#
my $a2wWatermarkTmp = new a2w::Text();
$a2wWatermarkTmp->setText("Confidential");
$a2wWatermarkTmp->setXPos(81); # X=81mm
$a2wWatermarkTmp->setYPos(204); # Y=204mm
$a2wWatermarkTmp->setAngle(60); # Rotation Angle = 60 degrees
$a2wWatermarkTmp->setColor(0xBDDBE7); # RRGGBB color value (BD=>189,
DB=>219, E7=>231 is light blue)
$a2wWatermarkTmp->setFont($a2wFontTmp); # set font

#---- Add Text to Page
$a2wPagePar->addText($a2wWatermarkTmp);
```

Das Script Beispiel ausführen

Sie verwenden die folgenden Kommandos in der Demo Stapeldatei um AFP2web mit der Scripting Facility zu starten:

Listing ***Demo Stapeldatei***

```
On Windows:  
afp2web.exe -q -c -doc_cold -sp: addWatermark.pm samples\insure.afp  
  
On Unix:  
./afp2web -q -c -doc_cold -sp: addWatermark.pm samples/insure.afp
```

Ergebnisse von addWatermark.pm

Dieses Skript erstellt PDF Dokumente mit einem eingefügten Text als Wasserzeichen auf jeder Seite. Wenn Sie das Skript-Beispiel mit einer Demoversion von AFP2web ausführen, so fügt die Demoversion von AFP2web den Text "Demo Version" als weitere Hintergrundgrafik hinzu.

addAnnotations.pm: Eine Hyperlink-Verknüpfung hinzufügen

addAnnotations.pm verstehen

In dieser Beschreibung konzentrieren wir uns auf die Subroutinen:

<i>Subroutine</i>	<i>Beschreibung</i>
sub initialize()	Initialisiert Attribute für die Darstellung der Hyperlink-Verknüpfung.
sub afp2web()	Haupteinstieg für AFP2web. Prüft ob die aktuelle Seite die erste Seite im Dokument ist und legt eine Hyperlink-Verknüpfung über einen Text an einer vorgegebenen Position.

Die Subroutine initialize() definiert die Eigenschaften für die Darstellung des Hyperlinks:

Listing***addAnnotations.pm - initialize()***

```

sub initialize(){

# Assuming width of a character as 2.53 millimeter
# It is used to calculate Hyper Link Rectangle Width as below
# Hyper Link Rect Width (in millimeters)=Hyper Link Text Length*$CharWidth
$CharWidth = 2.53;

# Hyper Link Rectangle Height values in millimeters ---
$HPLHeight= 2.6;

# Hyper Link Rectangle Border Width value in millimeters (approximately
1.0 Point)
#--- Set BorderWidth to zero if border is not needed.
$BorderWidth = 0.35;

#--- Hyper Link Action URL
$UrlText = "http://afp2web.com";

# Hyper Link Border color as RGB value (default color blue is assigned
here)

$HPLColor=0x000000FF; #---- Blue (0x00RRGGBB)
# $HPLColor=0x00FF0000; #---- Red
# $HPLColor=0x0000FF00; #---- Green
# $HPLColor=0x00FFFFFF; #---- White
# $HPLColor=0x00000000; #---- Black

}

```

Die Routine `afp2web()` verarbeitet jede Dokumentenseite. Ist die aktuelle Seite die erste Seite in einem Dokument, so wird ein Text an der vorgegebenen Position mit einem Hyperlink unterlegt. Die folgenden Zeilen zeigen, wie der Hyperlink als PDF Annotation definiert wird:

Listing***addAnnotations.pm - afp2web() - PDF Annotation vom Typ Hyperlink einfügen***

```

$a2wPagePar->addAnnotation( $CheckXPosTmp, #---- Upper left X position of
annotation box (in AFP page units)
$CheckYPosTmp, #---- Upper left Y position of annotation box (in AFP page
units)
$WidthTmp, #---- Width of annotation box (in AFP page units)
$HeightTmp, #---- Height of annotation box (in AFP page units)
$UrlText, #---- URL
$BorderWidthTmp, #---- Annotation border width
$HPLColor); #---- Annotation border color (Format=0x00BBGGRR)

```

Wir wiederholen nicht die Information, die Sie ausführlicher als Inline-Kommentare im Skript Beispiel finden. Dort finden Sie z.B. eine Beschreibung zu jedem Hyperlink Attribut.

Das Script Beispiel ausführen

Sie verwenden die folgenden Kommandos in der Demo Stapeldatei um AFP2web mit der Scripting Facility zu starten:

Listing *Demo Stapeldatei*

```
On Windows:
afp2web.exe -q -c -doc_col d -sp: addAnnotations.pm samples\insure.afp

On Unix:
./afp2web -q -c -doc_col d -sp: addAnnotations.pm samples/insure.afp
```

Ergebnisse von addAnnotations.pm

Sie finden das Konvertierungsergebnis standardmäßig im Unterverzeichnis /pdf. Die Hyperlink-Verknüpfung im PDF Dokument erscheint wie folgt:

Insurance Specialists Company will pay the benefits provider

We have issued this Policy to You in consideration of the payment of
 Your application. Your application is part of Your Policy.

Insured

Beth N McShine



<http://afp2web.com>

Policy Number

332-5767288-77

afp2xml.pm: Skript zur Umsetzung von AFP zu XML

Dieses Skript demonstriert, wie Sie aus AFP Daten XML Daten produzieren. Das geschieht in zwei Schritten: Sie extrahieren im ersten Schritt alle Information aus den AFP Daten, die Sie benötigen. Im zweiten Schritt erzeugen Sie die XML Elemente und Attribute. Die gültige XML Struktur wird mittels Document Type Definition (DTD) definiert und dient als Grundlage für die Erstellung der XML Daten.

Das Beispiel afp2xml.pm extrahiert Text und Indexdaten und weitere Informationen aus dem AFP Spool und bildet diese in einer XML Datei ab.

afp2xml.pm verstehen

Die folgenden Packages sind erforderlich:

Listing afp2xml.pm - Packages
<pre>use a2w: : Conf i g; use a2w: : Document; use a2w: : Font; use a2w: : I n d e x; use a2w: : Kernel ; use a2w: : L i n e; use a2w: : Medi umMap; use a2w: : NOP; use a2w: : Overl ay; use a2w: : Page; use a2w: : PSEG; use a2w: : Text; #---- Required Perl Modules for XML output use IO: : F i l e; use XML: : W r i t e r;</pre>

Das Skript verwendet das Package XML::Writer, eine einfaches Perl Modul zur Erstellung von XML Dokumenten. Sie erhalten dieses Modul auf der Website der Perl Comprehensive Network (www.cpan.org). Wir stellen dieses Modul in das Unterverzeichnis lib\XML\ der Perl Installation ab.

Eine typische Folge von Methoden zur Erstellung eines XML Dokuments ist wie folgt: my \$writer = new XML::Writer();\$writer->startTag('greeting', 'type' => 'simple');\$writer->characters("Hello, world!");\$writer->endTag('greeting');\$writer->end();

Für die Verwendung des Packages XML::Writer bitten wir Sie die offizielle Dokumentation auf der CPAN Website zu lesen. Dort finden Sie eine Referenz der folgenden Funktionen:

Funktion:	Beschreibung
<code>\$xmlDocWriter->xmlDecl</code>	Fügt die XML Deklaration an den Beginn eines XML Dokuments ein. Die Version ist stets ``1.0``.
<code>\$xmlDocWriter->doctype</code>	Fügt eine DOCTYPE Deklaration in das XML Dokument ein. Die Deklaration muss vor dem Wurzel-Element erfolgen.
<code>\$xmlDocWriter->comment</code>	Fügt ein Kommentar in das XML Dokument ein.
<code>\$xmlDocWriter->startTag</code>	Fügt den Start Tag in das XML Dokument ein. Als Argumente werden der Element-Name und Name/Werte-Paare für Attribute. Das Modul ersetzt Sonderzeichen '&', '<', '>', und '' in den Attributwerten mit vordefinierten XML Entitäten.
<code>\$xmlDocWriter->endTag</code>	Fügt ein Ende Tag für das Element in das XML Dokument. Das Ende Tag gilt für den jeweils zuletzt gesetzten Start Tag. Es muss zu jedem Start Tag ein Ende Tag gesetzt werden.
<code>\$xmlDocWriter->dataElement</code>	Den kompletten Inhalt eines Elements bestehend aus Zeichenfolgen ausgeben.
<code>\$xmlDocWriter->characters</code>	Zeichenfolge in ein XML Dokument einfügen. Alle Sonderzeichen '<', '>', und '&' im Argument \$data werden automatisch mit vordefinierten XML Entitäten ersetzt.
<code>\$xmlDocWriter->end</code>	Das XML Dokument abschließen. Stellt sicher, dass das Dokument genau aus einem Dokument-Element besteht und dass alle Start Tags mit Ende Tags abgeschlossen werden.

Die Subroutinen, die im Skript vorhanden sein müssen sind:

- `sub afp2web()` - der Haupteinstieg für AFP2web
- `sub initialize()`
- `sub initializeDoc()`
- `sub initializePage()`
- `sub finalizePage()`
- `sub finalizeDoc()`
- `sub finalize()`

In dieser Beschreibung konzentrieren wir uns auf die Subroutinen:

<i>Subroutine</i>	<i>Beschreibung</i>
sub initializeDoc()	Erstellt den Start-Tag für das Element Document.
sub afp2web()	Haupteinstieg für AFP2web. Erstellt Start-Tags für PageList und Page und erzeugt XML für Elemente auf dieser Seite.
sub finalizePage()	Erstellt den Ende-Tag für das aktuelle Element Page.
sub finalizeDoc()	Erstellt Ende-Tags für die Elemente PageList und Document

Die Subroutine initializeDoc() definiert das Wurzelement für das Dokument:

Listing

afp2xml.pm - initializeDoc() - Das Wurzelement für XML Dokument erstellen

```
#---- Generating FileHandle for XML Writer (needed since a2wDocumentPar-
>getSimpleFilename() not available from initializeDoc())
$xml DocIdFilename = $svOutputFilePath . $DocIdTmp . ".xml";
$xml DocOutput = new IO::File( ">" . $xml DocIdFilename );

#---- Generating XML Writer Object
$xml DocWriter = new XML::Writer(OUTPUT=> $xml DocOutput,
  NAMESPACES=> 1,
);

#---- Processing Instruction
$xml DocWriter->xml Decl ("ISO-8859-1");
$xml DocWriter->comment("Generated by $svScriptProc " . $Version . ", " .
  local time());
$xml DocWriter->doctype(' Document', "", ' afp2xml.dtd');

#---- Starting with root element
$xml DocWriter->startTag(' Document', ' Id' => $DocIdTmp, # Doc Id element
  ' Name' => $a2wDocumentPar->getName(),
  ' Spool' => $svSpoolFilename,
  ' PID' => $a2wDocumentPar->getPID(),
);
```

Die Subroutine afp2web() wird für jede Dokumentenseite aufgerufen. Hier werden die Start-Tags für die Elemente PageList und Page und die XML Daten für die Elemente auf dieser Seite erzeugt.

Listing***afp2xml.pm - afp2web()***

```

#---- Add Document Indexes to Index List
if ($PageId == 1){
  addIndexes( $a2wDocumentPar );
  $xml DocWriter->startTag(' PageLi st '); # <PageLi st>
}

#---- Starting with Element Page
$xml DocWriter->startTag(' Page', ' Id' => $PageId, # Page Id element
' Wi dth' => $a2wPagePar->getWi dth(),
' Hei ght' => $a2wPagePar->getHei ght(),
' Resol uti on' => $a2wPagePar->getResol uti on()
);

#---- Add Page Applied Medium Map Info
addPageMedi umMapI nfo();

#---- Add Page Applied Medium Overlay Li st
addPageMedi umOverl ayLi st();

#---- Add Page Included Overlay Li st
addI ncl udedOverl ayLi st( $a2wPagePar );

#---- Add Page Included PageSegment Li st
addI ncl udedPageSegmentLi st( $a2wPagePar );

#---- Add Page Indexes to IndexList of Page
addI ncl udedPageSegmentLi st( $a2wPagePar );

#---- Add NOPS to NOPList of Page
addPageNOPs();

#---- Add Text Objects to TextObje ctLi st of Page
addTextObje cts( $a2wPagePar );

#---- Add Line Objects to LineObje ctLi st of Page
addLi neObje cts( $a2wPagePar );

```

Wie im Skript-Beispiel das Modul XML::Writer für die Erstellung von XML Elementen verwendet wird sehen Sie am besten direkt in der Skript-Datei.

Das Skript Beispiel ausführen

Hinweis: Sie müssen für dieses Skript Beispiel Perl installieren. Beachten Sie bitte die Systemvoraussetzungen im Kapitel über die Installation von AFP2web.

Die folgenden Zeilen zeigen eine einfache Methode, mit der Sie weitere Verzeichnisse in Standard Suchpfad von Perl einfügen. Beachten Sie bitte: Sie müssen die Verzeichnisse entsprechend Ihrer Umgebung angeben. Wichtig: Diese Anweisung muss als erste Zeile in dem Skriptmodul stehen, den AFP2web als erstes lädt (in diesem Falle ist das afp2xml.pm):

Listing

Mit dem internen Perl Array @INC Verzeichnisse in den Perl Suchpfad einfügen

```
BEGIN {  
  unshift(@INC, ('C:\Perl\lib', 'C:\Perl\site\lib', 'C:\temp'));  
}
```

Der Befehl, mit dem Sie AFP2web starten und das Skriptmodul afp2xml.pm laden ist wie folgt:

Listing

Demo Stapeldatei

```
On Windows:  
FOR /F %i IN ( 'perl -e "{$,=';'; print @INC;}"' ) DO @SET PERLLIB=%i  
afp2web.exe -q -c -doc_cold -sp:afp2xml.pm samples\insure.afp  
  
On Unix:  
export PERLLIB=`perl -e "{$,=';'; print @INC;}"`  
./afp2web -q -c -doc_cold -sp:afp2xml.pm samples/insure.afp
```

Ergebnisse von afp2xml.pm

Für jedes AFP Dokument wird ein PDF Dokument und eine XML Dokument erstellt.

Die XML Darstellung des AFP Dokuments ist eine Textdatei ohne Einrückungen und daher nur schwer lesbar:

```

0      10      20      30      40      50      60      70
1  <?xml version="1.0" encoding="ISO-8859-1"?>
2  <!-- Generated by afp2xml.pm v1.0.0, Fri Aug 19 18:10:49 2005 -->
3  <!DOCTYPE Document SYSTEM "mhtDoc_V3.0.dtd">
4  <Document Id="1" Name="insure" Spool="samples/insure.afp" PID="s1qg"><Index
  .   " YPos="6176" CodedFont="" CharacterSet="C1DOGT10" CodePage="T1D0BASE" Type
  .   trikeover="no">If you are not satisfied with Your Policy, You may cancel it
  .   ents">Additional Benefits</Index><Index Name="Contents">Premium Summary</I
  .   h="-1" Weight="Bold" Slant="Medium" Underline="no" Strikeover="no">.....
  .   ace="GOTHIC" Height="100" Width="-1" Weight="Normal" Slant="Medium" Underl
  .   T1GI0361" Typeface="HELVETICA LATIN1" Height="100" Width="-1" Weight="Bold"
  .   -1" Weight="Normal" Slant="Medium" Underline="no" Strikeover="no">--</Text(
  .   h="-1" Weight="Bold" Slant="Medium" Underline="no" Strikeover="no">Part 1 .
  .   Height="100" Width="-1" Weight="Normal" Slant="Medium" Underline="no" Stril
  .   61" Typeface="TIMES NEW ROMAN LATIN1" Height="100" Width="-1" Weight="Bold"
  .   100" Width="-1" Weight="Normal" Slant="Medium" Underline="no" Strikeover="
  .   >2.</TextObject><TextObject XPos="4152" YPos="11407" CodedFont="" Character
5

```

Sie können einen XML Editor verwenden, um eine formatierte Ansicht auf das XML zu bekommen. Um dem XML Editor die Validierung zu ermöglichen, sind die DTDs für Dokument und Indexdaten im Unterverzeichnis /pdf abgestellt:

XML

Comment: Generated by afp2xml.pm v1.0.0, Fri Aug 19 18:10:49 2005

DOCTYPE Document

Document

- Id: 1
- Name: insure
- Spool: samples/insure.afp
- PID: s1qg

IndexList

- Index (2)

	Name	abc Text
1	Insured	Geoffrey R Stephens
2	Policy	324-1443255-11

PageList

- Page (3)

	Id	Width	Height	Resolution	MediumMap
1	1	12240	15840	1440	MediumMap Na...
2	2	12240	15840	1440	MediumMap Na...
3	3	12240	15840	1440	MediumMap Na...

A.3 Die PDF/A Unterstützung in AFP2web

Zur Unterstützung von PDF/A in AFP2web

AFP2web erzeugt Ausgabe in PDF Format, das dem Standard der Stufe PDF/A-1b entspricht. Der Standard wird definiert in ISO 19005-1. Document management — Electronic document file format for long-term preservation — Part 1: Use of PDF 1.4 (PDF/A-1) Reference number ISO 19005-1:2005(E)

PDF/A-konforme Dokumente sind selbstdefinierende PDF Dateien, die für die Langzeitarchivierung verwendet werden können. Der Standard basiert auf PDF Version 1.4 mit weiteren Einschränkungen in der PDF Funktionalität.

Adobe Systems Limited (www.adobe.com), der Hersteller dieser Technologie, formuliert - frei übersetzt - "[PDF/A-konforme Dateien] werden primär für die Archivierung verwendet. Da die Langzeitarchivierung das Ziel ist, muss das Dokument ausschließlich das beinhalten, was es im Laufe seiner erwünschten Existenz benötigt, um geöffnet und dargestellt zu werden. So können beispielsweise PDF/A-konforme Dateien nur Texte, Rasterbilder, Vektorobjekte enthalten, nicht aber Verschlüsselungen und Skripte. Auch müssen alle Schriften eingebettet werden, damit das Dokument geöffnet und in der Form betrachtet werden kann, die ursprünglich erzeugt wurde."

Zum Inhalt

Dieser Anhang ergänzt die AFP2web Version 3.x Benutzerhandbuch und Referenz. Er beschreibt wie Sie AFP2web Version 3.x verwenden, um PDF/A-konforme Ausgabedokumente zu erzeugen.

Wichtige Hinweise

AFP2web validiert nicht das erzeugte PDF/A. Es liegt in der Verantwortung des Nutzers von AFP2web die AFP2web Parameter korrekt zu definieren und solche Merkmale in den Dokumenten zu vermeiden, die für PDF/A nicht erlaubt sind (zum Beispiel die Referenzierung externer Schriften oder Transparenz).

INI Parameter und Kommandozeilenoptionen

Für die Ausgabe nach PDF/A-1b sind die folgenden Parameter relevant:

<i>INI Parameter</i>	<i>Kommandozeilen -option</i>	<i>Beschreibung</i>
OutputFormat=PDFA	-pdfa	Legt als Ausgabeformat PDF/A mit dem Level PDF/A-1b fest.
Strict=on	-strict	Wird AFP2web für die PDF/A Ausgabe ausgeführt, so müssen Sie den Parameter Strict=on verwenden. Dies bewirkt, dass AFP2web die Verarbeitung abbricht, wenn eine erforderliche Ressource (Schrift oder AFP Ressource) nicht gefunden werden kann.
CMYKtoRGB=on		<p>Sorgt dafür dass alle Farben in einem einzigen Farbraum definiert sind. AFP2web konvertiert alle Farben zu RGB.</p> <p>Hinweis: AFP2web verwendet das Farbprofil ICC sRGB IEC61966-2-1 zur Definition des OutputIntent.</p> <p>Wichtig: Verwenden Sie nicht die Kommandozeilenoption -nocmyktorgb, da diese den INI Parameter überschreibt.</p>

Schriften

Für die Ausgabe nach PDF/A müssen die Schriften in der PDF Datei eingebettet werden. AFP2web verwendet Zuordnungstabellen in der Datei mapping.def. Diese Datei hat den Abschnitt [CHARSET RENDERING] in dem für jedes AFP Character Set definiert werden kann, wie die Schrift zu verarbeiten ist. Hier kann entweder die Verwendung der originalen AFP Schrift oder einer Ersatzschrift vorgegeben werden.

Für die Ausgabe nach PDF/A empfehlen wir generell, Schriften nicht zu ersetzen.

<i>Konfigurationseinstellung</i>	<i>Erläuterung</i>
mapping.def, Abschnitt [CHARSET RENDERING], Rendering flag	<p>Bitte verwenden Sie - wenn erforderlich - als Rendering Flag entweder:</p> <p>0 (Standard für AFP Schriften)</p> <p>2 um eine Ersatzschrift einzubetten. Für PDF/A unterstützt AFP2web nur Type 1 Schriften.</p> <p>Wichtig: Verwenden Sie als RenderingFlag nicht den Wert 1, um die Schrift zu referenzieren, da diese Einstellung für PDF/A nicht erlaubt ist.</p>

Metadaten im XMP Format

AFP2web generiert Metadaten im XMP Format automatisch, bettet diese in der PDF als Stream-Objekt ein und referenziert diese mit Hilfe eines "Metadata" Eintrags in der Catalog Dictionary.

AFP2web ordnet Metadaten wie folgt zu:

<i>INI Parameter</i>	<i>XMP Metadata</i>		<i>Werte</i>
	<i>Feld</i>	<i>Typ</i>	
Title	dc:title	Text	Title laut INI Parameter
Subject	dc:subject	Text	Subject laut INI Parameter
Keywords	pdf:keywords	Text	Keywords laut INI Parameter
-----	dc:creator	Text	AFP2web Version ?? [Built for <OS> on <zeitstempel>]
-----	xmp:CreatorTool	Text	AFP2web Version ?? [Built for <OS> on <zeitstempel>]
-----	pdf:Producer	Text	AFP2web Vx.x [<OS name> on <zeitstempel>]
-----	xmp:CreateDate	Date	Datum und Zeit der PDF Erstellung

Hinweis: AFP2web generiert auch Metadaten für Outline Fonts.

Merkmale, die für PDF/A nicht zu verwenden sind

Die folgende Tabelle führt die Merkmale auf, die nicht mit dem Standard PDF/A konform sind. Als Lösung werden die Maßnahmen beschrieben, die Sie bei der Verwendung von AFP2web für PDF/A ergreifen:

<i>Einschränkung</i>	<i>Lösung</i>
Keine Hyperlinks zu externe Dateien.	Annotationen in PDF mit der Hyperlink-Funktionalität sind nicht möglich. Vermeiden Sie bitte die Scripting Facility "addAnnotation" Schnittstelle von "a2w::Page".
Scripting Facility, a2w::Page->add-Text	Verwenden Sie nicht die Scripting Facility "add-Text" Schnittstelle von "a2w::Page". Diese Funktion bettet die Schrift nicht ein.
Verschlüsselung (Encryption)	PDF Sicherheitseinstellungen sind nicht erlaubt. Verwenden Sie daher keine Sicherheitseinstellungen (in dem INI Parameter PDFSecurity oder der Kommandozeilenoption -ps)
Transparenz	Verwenden Sie keine Transparenz. Bei der Ausgabe nach PDF/A wird AFP2web die Transparenz entfernen.
Wasserzeichen	Fügen Sie mit AFP2web keinen Text als Wasserzeichen ein. Verwenden Sie weder den INI Parameter Watermark noch die Kommandozeilenoption -wm.

A.4 Frequently Asked Questions (FAQs)

E088: Scripting Facility Error (rc=-998): Unable to parse the file

Problem:

AFP2web bricht ab mit der Fehlermeldung:

```
Can't locate a2w/MyModule.pm in @INC (@INC contains: listofPerl-  
paths.) at NOPs.pm line 'n'.
```

```
...  
E088: Scripting Facility Error (rc=-998): Unable to parse the file
```

Lösung:

Überprüfen Sie Ihre Einstellungen für Perl. Siehe hierzu die Hinweise in dem AFP2web und Scripting Facility Benutzerhandbuch und Referenz.

Stichwortverzeichnis

Symbols

-? 151

A

a2w Config	220
a2w Document	223
a2w Font	231
a2w Index	242
a2w NOP	260
a2w Overlay	263
a2w Page	271
a2w Text	291
addAnnotations.pm	382
addBookmarks.pm	376
addWatermark.pm	380
AFP	
Kurvenglättung	148
AFP Index	161
AFP Ressourcen	47, 48, 158
AFP Ressourcendatei	158
AFP Ressourcenpfad	136
AFP2web	89
Anwendungsszenarios	29
Dokumente auswählen	52
Eingabeformate	26
Funktionsübersicht	25
Installation	37
Kernkomponenten	27
Konfiguration, Überblick	43
Konfigurationsdateien für Schrift	69
Parameter für die Kontrolle	51
PDF Lesezeichen	61
Produktarchitektur	28
Schriftverarbeitung	75
SDK Option	28
starten AFP2web via CLI	50
stoppen AFP2web via CLI	50
Wasszeichen als Hintergrundtext	62
wichtige Hinweise	19
AFP2web neue Merkmale	13
afp2web()	97
afp2web.ini	
Abschnitt AFPCOLORTABLE	142
Abschnitt FORMDEF	141
Abschnitt für FTP Schnittstellen	143
Abschnitt settings	
AFPComplianceLevel	120
AFPFontPath	120
Author	122

Autosplit	120
CharSetExt	120
CMYKTORGB	121
CodedFontExt	121
CodePage	121
CodePageDefaultChar	121
CodePageExt	121
Color	122
CpPath	122
CurveSamplingFactor	122
DocumentCount	122
EndingDocument	123
EndingPage	123
ExceptionLoggingLevel	123
ExtFontPath	123
FileCreationMode	123
FilenamePattern	124
FormatType	125
FormDefExt	125
FormDefPath	125
GenerateUniqueFile	126
IndexFormat	126
IndexPath	126
IndexRecord	126
InputFormat	127
JPEGQuality	127
Keywords	127
LaunchPreview	127
Licensee	128
Logging	128
LoggingFont	128
LoggingLevel	129
LogPath	130
MaxCols	130
MemoryOutputStream	130
OutputFilePath	130
OutputFormat	130
OutputSize	131
OverlayExt	131
OverlayPath	131
PageOutput	131
PageRotation	132
PageSegExt	132
PageSegmentPath	132
PDFBookmark	132
PDFDocLimits	133
PDFSecurity	133
PDFUIOptions	134
PDFWinOptions	135
PrintableLineWidth	136
Protocol	136
Quietmode	136

Resolution	136
ResPath	136
SaveOCDataOnDisk	136
ScriptArgument	137
ScriptProcedure	137
ScriptUnitBase	137
SerialNr	137
SkipObjectSize	138
SkipPage	138
StartingDocument	138
StartingPage	138
Statistic	138
Strict	138
Subject	139
TempPath	139
Title	139
Watermark	139
Funktion und Struktur	119
afp2web.pm	214
afp2xml.pm	385
AFPComplianceLevel	120
AFPFontPath	120
ALL	123
Anzeige der Konvertierungsergebnisse	127
Auflösung	155
Ausgabe	
ASCII Seitenbegrenzungen	153
empfohlene Werte für die ASCII Formatierung	151
Formate	123, 130, 145
TIFF	125
Indexdaten Zielverzeichnis	161
Kompression für JPEG	127
Lesezeichen in PDF	132
Logging	
Schalter	152
Logging Schalter	149, 152
Meldungen an die Konsole unterdrücken	136
Muster für den Dateinamen	124
Pfad für Ausgabedateien	130
Pfad für Log-Dateien	130, 153
Standard Index Format	126
Verarbeitungsstatistik	138
Verzeichnisse für die Ausgabe	56
Ausgabe einer Datei pro Seite für TIFF	
TIFF	
Ausgabe einer Datei pro Seite für TIFF	131
Ausgabe in Unterverzeichnisse	122, 149
Ausgabe nach stdout	159
Author Eigenschaft für PDF	122
Autosplit	120
autosplit.pm	335
B	
Barcodeformate	59

C

Character Sets	
Zuordnung zu Code Pages	179
CharSetExt	120
-clr	148
CMYKTORGB	121
Code Page Dateien	179
Einführung	72
Code Page Default	121
Code Page Default Zeichen	121
Code Page Zuordnung	179
Code Pages Pfad	148
CodedFontExt	121
CodePage	121
CodePageDefaultChar	121
CodePageExt	121
-cp	148
CpPath	122
-csf	148
CurveSamplingFactor	122

D

Dateien	
ASCII code pages	122
Dateilokation	
AFP Fontressourcen	147
AFP Overlay Ressourcen	131
AFP Page Segments	132
AFP Ressourcen	136
Ausgabedateien	130
Code Page	148
Formdefs	125
Indexausgabe	126
Schriften für PDF	123
temporäre Arbeitsdatei	139
Dateilokationen	
AFP Ressourcen	158
AFP Formdefs	150
AFP Index	161
AFP Overlay Ressourcen	154
AFP Page Segment Ressourcen	156
AFP Ressourcendatei	158
Ausgabedateien	154
Indexdaten Zielverzeichnis	161
INI-Datei	151
Pfad zur afp2web.ini	151
Schriften für PDF	151
temporäre Arbeitsverzeichnis	159
Dateinamen festlegen	124
Dateinamenerweiterung	
AFP Character Sets	120
AFP Code Pages	121
AFP Coded Fonts	121
AFP Formdefs	125
AFP Overlay Ressourcen	131

AFP Page Segments	132
Dateinamenmuster	150
Dateinamenserweiterungen	
Überblick	47
-dc	149
-dcp	149
DOC	123
DOC_COLD	123
DOC_INDEX	123
DocumentCount	122
dumpIniParms.pm	331
dumpMediumMaps.pm	358
dumpNOPs.pm	362
dumpPageTextObjs.pm	345

E

-ed	149
eine Datei pro Dokumentseite für TIFF ausgeben	154
Eingabe	
AFP Index	161
Formate	127, 146
Eingabe aus stdin	159
Eingabe und Ausgabe via stdin und stdout	159
-ell	149
EndingDocument	123
EndingPage	123
error messages	
MHTIMG Library messages	313
ExceptionLoggingLevel	123
ExtFontPath	123
eyecatcher.pm	366

F

Farbe	
CMYK zu RGB für IOCA	121, 142, 153
IOCA CMYK zu RGB konvertieren	148
Verarbeitungsschalter	122, 147
-fd	150
-fdp	150
Fehlerbehandlung	
Abbruch bei fehlender Ressource	138,
159	
Abbruch erzwingen bei fehlender Ressource	63
Fehlermeldungen zu AFP2web	300
FileCreationMode	123
FilenamePattern	124
finalize()	98
finalizeDoc()	98
finalizePage()	98
-fnpt	150
Fontmetriken	17
Formate	

Auflösung für Rasterformate	136
Ausgabeformate	130, 145, 146
Eingabeformate	14, 127, 146
Index	126
Kompression für JPEG	127
PDF	14
PDF/A	14
TIFF Ausgabe	125
TIFF Komprimierung	147
Überblick der Formate	26

FormatType	125
Formdef Pfad	150
FormdefExt	125
FormDefPath	125
Formdefs	141
Formdefs angeben	48
-fp	151
Frmate	
TIFF Verbesserungen	15
Funktion	44

G

GenerateUniqueFile	126
--------------------------	-----

H

-h	151
HostName	143

I

-iASCII	151
-if	151
Images	
Optimierung	58
Verarbeitung von Included Objects	57
Index	
AFP2web Standardverarbeitung	54
Ausgabeformat CSV	126
Ausgabepfad	126
Benutzerdefinierte Indexdaten im AFP Datenstrom	100
Optionen	54
Standard AFP Index	100
Standard Format	126
Verarbeitung mit der Scripting Facility	100
Verarbeitungsschalter	123
IndexFormat	126
IndexPath	126
IndexRecord	126
INI Datei	
Parameter Überblick	111
INI-Datei	44
INI-Datei in der Kommandozeile angeben	151
initialize()	97

initializeDoc()	97
initializePage()	97
InputFormat	127
IOCA Verarbeitung	57, 58

J

JPEGQuality	127, 152
-jq	152

K

Keywords	127
Kommandozeilenoption	
?	151
allgemeine Anmerkungen	145
-ap	147
Ausgabeformate	145, 146
-bm	147
-c	147
-clr	148
-cp	148
-csf	148
-dc	149
-dcp	149
-ed	149
Eingabeformate	146
-ell	149
-fd	150
-fdp	150
-fnpt	150
-fp	151
-h	151
-iASCII	151
-if	151
-jq	152
-l	152
-lf	152
-ll	152
-lp	153
-mc	153
-nocmyklogb	153
-op	154
-os	154
-ovp	154
-p	154
-plw	155
-po	154
-pp	154
-pr	155
-ps	155
-psp	156
-pui	156
-pv	157
-pwn	157
-q	157
-r	158

-rf	158
-rp	158
-s	158
-sa	158
-sd	158
-si	159
-so	159
-sod	159
-sp	159
-std	159
-Strict	159
TIFF Komprimierung	147
-tp	159
-u	160
-v	160
-vall	160
-wm	160
-xf	161
-xp	161

Kommandozeilenoptionen	
Überblick	111
Kompression für JPEG	127
Komprimierung für TIFF	147
Konsole	
ASCII, empfohlene Werte anzeigen	151
Hilfe anzeigen	151
Meldungen unterdrücken	136, 157
Konvertierung von PDF zu AFP	13

L

-l	152
LaunchPreview	127
Lesezeichen in PDF	61, 132, 147
-lf	152
Licensee	128
Linienstärke in AFP	155
-ll	152
Loggin	
Schalter	128
Logging	128
die Verwendung von LOG-Dateien	64
Protokoll	154
Verarbeitungsprotokoll ausgeben	136
Verarbeitungsstatistik	158
logging	
flags	123
Schriftinformation	181
Logging Schalter	152, 154
LoggingFont	128
LoggingLevel	129
Logging-Schalter	149
LogPath	130, 153
-lp	153

M

Maas Fontmetirken	17
mapping.def	
Abschnitt AFPFONT	176
Abschnitt CHARSET	170
Abschnitt CHARSET RENDERING	166
Abschnitt CODEPG	177
Abschnitt FGID	171
Abschnitt FONTSUFFIX	172
Abschnitt PDFFONT	173
Abschnitt UNIXFONT	175
Abschnitt WINFONT	174
Abschnitt XFONT	169
Funktion	71
Überblick	163
MaxCols	130
-mc	153
Meldungen	
AFP2web Fehlermeldungen	300
ExceptionLoggingLevel	123
Log-Ausgabe	130
Log-Schalter	128, 129
Statistiken	138
unterdrücken	136
messages	
MHTIMG Library messages	313
migration	
Scripting Facility modules to Version 3.x ...	321
Migration nach AFP2web Version 3.x	21
Miniaturansicht ausgeben	131, 154

N

neue Merkmale	13
a2w Kernel	248
a2w Line	250
a2w MediumMap	257
a2w PSEG	269
Scripting Facility API	
a2w Kernel	248
a2w Line	250
a2w MediumMap	257
a2w PSEG	269
-nocmyktob	153

O

-op	154
optimization flag	136
Option für die Optimierung	159
-os	154
OutputFilePath	130
OutputFormat	130
OutputSize	131
OverlayExt	131

OverlayPath	131
-ovp	154

P

-p	154
PageOutput	131
PageRotation	132
PageSegExt	132
PageSegmentPath	132
Password	143
paths	
AFP Page Segments	132
AFP Ressourcen	136
PDF Eigenschaften	
Author	122
Keywords	127
Licensee	128
PDFSecurity	133
Subject	139
Title	139
PDF Sicherheitseinstellungen	155
PDF Speicherreservierung	133
PDF Viewer Einstellungen	156, 157
PDF Viewereinstellungen	134, 135
PDF/A Unterstützung in AFP2web	391
PDFBookmark	132
PDFDocLimits	133
PDFSecurity	133
PDFUIOptions	134
PDFWinOptions	135
Pfad	
AFP Fontressourcen	147
AFP Formdefs	150
AFP Index	161
AFP Overlay Ressourcen	131
AFP Ressourcen	158
AFP Ressourcendatei	158
afp2web.ini	151
Ausgabedateien	130
Code Page	148
Indexausgabe	126
Indexdaten Zielverzeichnis	161
INI-Datei	151
Schriften für PDF	123, 151
temporäre Arbeitsdatei	139
Pfad zu AFP Ressourcen	45
Pfade	
AFP Overlay Ressourcen	154
AFP Page Segment Ressourcen	156
ASCII Code Pages	122
Ausgabedateien	154
Formdefs	125
temporäre Arbeitsverzeichnis	159
-plw	155
-po	154
-pp	154
-pr	155
PrintableLineWidth	136

Protocol	136	Skriptauswahl	159
-ps	155	Spooldateitrennung in Dokumente und Seiten	101
-psp	156	Variablen Gültigkeitsbereiche	99
-pui	156	Verarbeitungsschalter	123
-pv	157	Verwendung	103
-pwn	157	Scripting Facility API	
Q		a2w Config	220
-q	157	a2w Document	223
Quietmode	136	a2w Font	231
R		a2w Index	242
-r	158	a2w NOP	260
Resolution	136	a2w Overlay	263
ResPath	136	a2w Page	271
-rf	158	a2w Text	291
Rotation	158	afp2web.pm	214
-rp	158	Referenz	213
S		Scripting Facility Beispiele	
-s	158	addAnnotations.pm	382
-sa	158	addBookmarks.pm	376
SaveOCDataOnDisk	136	addVWatermark.pm	380
Schrift		afp2xml.pm	385
Einführung in die Verarbeitung	67	autosplit.pm	335
Funktion der mapping.def	71	dumpIniParms.pm	331
Konfigurationsdateien	69	dumpMediumMaps.pm	358
LOG-Datei	181	dumpNOPs.pm	362
Log-Schalter	128, 129	dumpPageTextObjs.pm	345
Pfad zu Ersatzschriften	123	eyecatcher.pm	366
temporäre Arbeitsdatei	139	sortPageTextObjs.pm	352
Schriften	13	Scripting Facility Parsingereignisse	96
AFP Schriften Namenskonvention	192	Scripting Facility Schnittstelle	95
AFP2web konfigurieren	45	afp2web()	97
Logging Schalter	152	finalize()	98
Pfad zu AFP Fontressourcen	147	finalizeDoc()	98
Verarbeitungsmodi in AFP2web	75	finalizePage()	98
ScriptArgument	137	initialize()	97
Scripting Facility		initializeDoc()	97
Anwendungsszenarios	91	initializePage()	97
Argumente	137, 158	ScriptProcedure	137
Dokumentaushabe im Speicherpuffer	108	ScriptUnitBase	137
Enkodierung	99	-sd	158
Indexverarbeitung	100	Seitenorientierung	132
Interaktion mit AFP2web	94	Seitenrotation	158
Koordinaten	99	SerialNr	137
Maßeinheit	137	-si	159
Methoden im Überblick	205	SkipObjectSize	138
migrating to Version 3.x	321	SkipPage	138
neue Merkmale	15	-so	159
Perl Einstellungen	106	-sod	159
Routinen im Überblick	198	sortPageTextObjs.pm	352
Skript	137	-sp	159
		Speicherreservierung für PDF Verarbeitung	133
		Spooldatei	
		Auswahl der Dokumentseiten	154
		Dokumenttrennung mit der Scripting Facility	101
		erste Seite zu verarbeiten	138
		erstes Dokument auswählen	158
		erstes Dokument zu verarbeiten	138
		Kriterium für Leerseiten	138

Leere Seiten ignorieren	138
letzte Seite	123
letztes Dokument	123, 149
Standard Code Page	149
Standard Formdef	150
StartDir	143
StartingDocument	138
StartingPage	138
Statistic	138
Statistikreport	158
-std	159
-Strict	159
Strict	138
Subject	139

T

temporäre Arbeitsdatei	139
temporäre Arbeitsverzeichnis	159
TempPath	139
TIFF Komprimierung	147
Title	139
-tp	159

U

-u	160
Überblick	47, 95
UserName	143

V

-v	160
-vall	160
Versionsnummer aller AFP2web Komponenten	160
Versionsnummer von AFP2web	160
Viewer für Konvertierungsergebnisse starten	157

W

Wasserzeichen	160
Wasserzeichen als Hintergrundtext	62
Watermark	139
-wm	160

X

-xf	161
-xp	161

