StreamServe™ Persuasion™

StreamServe Persuasion SP5 AFPIN

User Guide

Rev A

StreamServe Persuasion SP5 AFPIN User Guide
Rev A

StreamServe Web Site
http://www.streamserve.com

Contents

The AFPIN filter

The AFPIN filter converts AFP input to Layout eXchange Format (LXF), and enables Streamserver to identify and extract AFP formatted input documents. Typical usage scenarios are to:

- Recognize pages and read formatted text from the AFP document using PreformatIN.
- Convert AFP documents to other print formats, such as PCL and PostScript.
- Convert AFP documents to archival formats, such as PDF and TIFF, and re-use indexing information.
- Convert AFP overlays to LXF overlays for migration to StreamServe environment.
- Convert AFP documents and hide or replace, for example, OMRs and barcodes.
- Store AFP document in the post-processor repository using AFP document metadata. The AFP documents can be sorted and included in the same envelope as other stored documents.

The AFP conversion may result in large LXF documents, depending on the complexity of your AFP data stream and PreformatIN design. The large LXF documents may affect Project performance, depending on factors such as throughput, operating environment and hardware.

AFP reference

You must have knowledge of the AFPDS format and AFP terminology to use the AFPIN filter.

See also http://www.afpconsulting.com/afpsupportctr/afptechlinks.htm

Notes about the AFPIN filter

Fonts

AFP fonts are always mapped to TrueType or Type1 fonts, that are available to the StreamServer.

Pagedef

AFP Pagedef is not supported.

Formdef

N-Up printing is not supported.

The following is supported from the AFP Formdef:

- sheet size
- duplex
- overlays
- trays and bins

Overlays

OGL source files must be converted into AFP overlays.

Images

Embedded images (TIFF, JPEG, BMP, EPS, etc.) are not supported.

Bitmap patterns from IM image and IOCA are replaced with boxes filled with solid color of the respective shade percentage.

Graphics

GOCA markers are not supported.

Supported barcodes

The following barcode types are supported:

AFP barcode	Corresponding StreamServe barcode names
Australia Post Bar Code	Austpost
Code 128	Code 128
CODE 39	Code 39
Data Matrix	Datamatrix
EAN-13	EAN-13
EAN-8	EAN-8
Industrial 2-of-5	Industrial 2 of 5
Interleaved 2-of-5	Interleaved 2 of 5
Matrix 2-of-5	Matrix
MaxiCode	Maxicode
MSI	MSI
PDF417	PDF417
POSTNET	Postnet
RM4SCC	Royal Mail
UPC/CGPC Version A	UPC-A
UPC/CGPC Version E	UPC-E

Configuring the AFPIN filter

The procedure for configuring the AFPIN filter is the same as for all other types of filter chain filters. See the *Design Center* documentation for information about filter chains.

Use the filter chain editor to configure the AFPIN filter with settings for a specific project. To specify defaults that can be used by multiple projects you use a settings file. For more information see *AFPIN filter settings* on page 17.

Font mapping

AFP fonts can not be read by the StreamServer and must be mapped to TrueType and Type1 fonts. You must create a font mapping file where you specify how to map the fonts.

A sample of a font mapping file is included in the AFPIN filter installation and located in the following directory:

`<installation_directory>/Common/modules/filters/AFPINfont.map`

The sample file contains mappings for AFP character sets, typefaces, coded fonts, FGID (Font Global Identifier), CPGID (Code Page Global Identifier) and code pages.

For mapping file syntax descriptions, see:

- *Character set mapping* on page 8
- *Typeface mapping* on page 10
- *Coded font mapping* on page 11
- *FGID mapping* on page 11
- *Code page mapping* on page 12
- *CPGID mapping* on page 12

To create a font mapping file

1. Copy the sample file `AFPINfont.map`.
2. Enter the keywords and mapping values according to the syntaxes. Use tab to separate the columns.
3. Save the file and add it to a resource set connected to the Platform.

Character set mapping

To map AFP character sets to TrueType fonts, use the following syntax:

`<AFP charset> CHARSET <font> [style] [size]`

where font style and size are optional.

Note: Style and size specified in the AFP character set resource overrides the settings in the font mapping file.

Example 1 *Character set mapping*

```
// Charset mapping
C0H200A0      CHARSET       Arial                      11.0
C0H30080      CHARSET       Arial      Italic           8.0
C0H40070      CHARSET       Arial      Bold             7.0
C0H500B0      CHARSET       Arial      Bold,Italic     12.0
```

Example 2 *Character set mapping when style and size are specified in the AFP character set resource*

```
// Charset mapping
C0H200A0      CHARSET       Arial
```

Forced character set mapping

If you want mapped fonts to always be bold or italic, or to never be bold or italic, you use the following keywords:

- `NOT_BOLD` – Mapped fonts will always be plain in the output.
- `USE_BOLD` – Mapped fonts will always be bold in the output.
- `NOT_ITALIC` – Mapped fonts will always be plain in the output.
- `USE_ITALIC` – Mapped fonts will always be italic in the output.

Use the following syntax in the charset mapping:

<AFP charset> CHARSET *<font>* *<keyword>* [*size*]

Example 3 *Forced character set mapping*

In this case, mapped font will always be bold independent on flags in found AFP charset C0HL05GP.

```
// Charset mapping
C0HL05GP      CHARSET       Arial    USE_BOLD     5.0
```

Forced font size mapping

You can use "`!`" to override the font size specified in the AFP charset resource.

Example 4 *Forced font size mapping*

In this case, the mapped font size will always be 10.0 independent on flags in found AFP charset COHL10GP.

```
// Charset mapping
COHL10GP      CHARSET       Arial   !10.0
```

Matching multiple character sets

Enter a question mark in the character set name to match multiple character set names. Any character in the position of the question mark is ignored.

Example 5 *Mapping matching multiple character sets*

In this example `C?H500B0` matches C0H500B0, C1H500B0, C2H500B0 etc.

```
// Charset mapping
C?H500B0      CHARSET       Arial      Bold,Italic     12
C?H30080      CHARSET       Arial      Italic           8
```

Specifying font size for multiple character sets

Enter the percentage sign in the character set name to specify the font size according to IBM font naming standard. The font size is derived from the character in the position of the percentage sign, according to the following rules:

1 = 1.0, 2 = 2.0, 3 = 3.0, …, 9 = 9.0, 0 = 10.0, A = 11.0, B = 12.0, C = 13.0 ...

Note: A font size specified in the AFP character set resource overrides the setting in the font mapping file.

Example 6 *Specifying fonts size for multiple character sets*

In this example `C?H500%0` matches, for example, C1H500B0. The font size is derived from the B and set to 12.

```
//Charset mapping
C?H500%0      CHARSET       Arial      Bold,Italic
C?H300%0      CHARSET       Arial      Italic
```

Typeface mapping

You can map AFP character sets to TrueType fonts using typeface mapping. Use the typeface name from the font descriptor in the AFP character set, and the following syntax:

`<AFP typeface name> TYPEFACE <font>`

Example 7 *Typeface mapping*

```
// Typeface mapping
TIMES ROMAN         TYPEFACE          Times New Roman
HELVETICA           TYPEFACE          Arial
COURIER LATIN1      TYPEFACE          Courier New
```

Forced typeface mapping

To map plain AFP typefaces to bold or italic, or bold and italic AFP typefaces to plain, you can use the following keywords:

- NOT_BOLD – Mapped AFP typefaces will always be plain in the output.
- USE_BOLD – Mapped AFP typefaces will always be bold in the output.
- NOT_ITALIC – Mapped AFP typefaces will always be plain in the output.
- USE_ITALIC – Mapped AFP typefaces will always be italic in the output.

Use the following syntax in the typeface mapping:

```
<AFP typeface name> TYPEFACE <font>  <keyword>
```

Example 8 Forced typeface mapping

In this case, AFP typefaces in Windings will always be plain in the output, even if original AFP typefaces are bold.

```
// Typeface mapping
WINGDINGS          TYPEFACE         Wingdings         NOT_BOLD
```

Coded font mapping

Use the following syntax to map coded fonts to a specific AFP code page with a specific AFP character set:

```
<coded font> CODEDFONT <AFP code page> <AFP character set>
```

Example 9 Coded font mapping

```
// Coded font mapping
X0A0558C     CODEDFONT        T1V10273       C0A05580
X0A0756C     CODEDFONT        T1V10273       C0A05560
```

FGID mapping

Use the following syntax to map FGIDs (Font Global Identifiers) to TrueType fonts:

```
<fgid> FGID <font> [style]
```

Note: The <*fgid*> must be a decimal integer.

Example 10 FGID mapping

```
// FGID mapping
05580      FGID       Arial       Bold
```

Code page mapping

Use the following syntax to map the AFP code pages to code pages used by the StreamServer:

`<AFP code page> CODEPAGE <code page>`

The code pages used by the StreamServer are found in the Design Center, for example, **Tools** > **Design Center Settings**.

Example 11 Code page mapping

```
// Codepage mapping
T1V10273        CODEPAGE        IBM CP 273
T1D1STMT        CODEPAGE        IBM CP 500
T1V10500        CODEPAGE        IBM CP 500
```

CPGID mapping

Use the following syntax to map CPGIDs (Codepage Global Identifiers) to code pages used by the StreamServer:

`<cpgid> CPGID <code page>`

Note: The <*cpgid*> must be a decimal integer.

The code pages used by the StreamServer are found in the Design Center, for example, **Tools** > **Design Center Settings**.

Example 12 CPGID mapping

```
// CPGID mapping
02819         CPGID         IBM CP 273
```

GCGID mapping

In a standard scenario where the AFP input only contains standard GCGIDs (Graphic Character Global Identifiers), the GCGIDs are automatically converted to Unicode value.

If the AFP input contains unavailable GCGIDs, you must specify a GCGID mapping table and/or specify GCGID generic masks. See *Creating a GCGID mapping file* on page 13 and *Specifying GCGID generic masks* on page 13.

Creating a GCGID mapping file

Use the following syntax in the GCGID mapping file:

<Unicode> <GCGID>

Example 13 The GCGID mapping file UserGCGID.txt

```
U0020        SX010000
U0021        SP0200X0
U0022        SP0400X0
U0023        SM0100W0
U0025        SM02000Y
```

To create a GCGID mapping file

1. Open a text editor.
2. Enter the mappings according to the syntax and use `TAB` to separate the columns.
3. Save the file, and add it to a resource set connected to the Platform.

Specifying GCGID generic masks

For GCGIDs that have been generated from Unicode values, you can use masks to derive the Unicode value. The masks must have the following format:

```
%[width]type
```

where

- `%` indicates the start of the Unicode value to derive.
- `width` specifies the maximum number of characters to read.
- `type` specifies the data type that is expected.

The following types can be used:

Type	Qualifying input
d	Decimal integer
x	Hexadecimal integer
u	Unsigned decimal integer
o	Octal integer

The GCGID masks must be separated by semicolons.

Example 14 GCGID masks

```
UD0%05d;U000%04x;UD0%05u;UO%06o
```

`UD0%05d` matches GCGIDs that begin with UD0. The following five characters are interpreted as a decimal number.

`U000%04x` matches GCGIDs that begin with U000. The following four characters are interpreted as a hexadecimal number.

`UD0%05u` matches GCGIDs that start with UD0. The following five characters are interpreted as unsigned decimal integers.

`UO%06o` matches GCGIDs that start with U0. The following six characters are interpreted as octal integers.

Extracting NOP records

You can extract NOP records from different levels in the AFP input data. The NOP records are converted to variable values that can be used for further processing.

In the filter chain editor you use the following arguments to specify what NOP records to extract:

[*apf_level*]:[*starts_with*]:<*property_name*>[;]

In the settings file you use the same arguments with the `AFP_NOP_MASKS` keyword.

Note: If multiple NOP records match your arguments, the variable will contain the value of the last processed NOP record.

NOP extracting arguments	
apf_level	The level where the NOP is extracted. If not specified, NOPs for all levels are extracted. Available levels are `BGN` and `BPG`.
starts_with	Initial pattern of the NOP value to extract. If not specified, all available NOPs are extracted.

property_name	Name of the variable that will contain the NOP. You can not use the following as variable names. The names are restricted to page description in the LXF file: • PAGEDUPLEX • PAGETRAY • PAGEBIN • PAGEORIENTATION • PAGEMEDIA • PAGEWIDTH • PAGEHEIGHT • MEDIUMMAP • MEDIAORIENT • AFP_BNG_NAME

Example 15 Extracting a NOP record at BNG level

In this example, NOP records at BNG level are extracted. The last processed NOP is stored in the variable `AFP_NOP`.

```
BNG::AFP_NOP
```

Example 16 Extracting a specific NOP record

In this example, NOP records starting with `333` at BNG level are extracted. The last processed NOP is stored in the variable `AFP_NOP`.

```
BNG:333:AFP_NOP
```

Example 17 Extracting multiple NOP records at any level

In this example, NOP records located at any level starting with `222` or `333` are extracted. The last processed record starting with `222` is stored in the `AFP_NOP1` variable and the last processed record starting with `333` stored in the `AFP_NOP2` variable.

```
:222:AFP_NOP1;:333:AFP_NOP2
```

Capturing and reusing AFP objects

To capture AFP objects from the AFP input data, for example TLEs, you must analyze the input data to find the names of the objects.

You use variables and scripting to capture the values of the objects.

Example 18 Capturing TLE values

In this example, the AFP name of the TLE is PageInfo.

```
$var=$PageInfo
```

Medium Map variables

When the AFP input document uses Medium Maps, the AFPIN filter extracts information from the currently used Medium Map, and adds this information to page variables. The variables are described in the table below.

Medium Map variables	
MEDIUMMAP	The name of the invoked Media Map.
MEDIAORIENT	The media orientation: • PORTRAIT • LANDSCAPE • PORTRAIT90 • LANDSCAPE90
PAGETRAY	The tray number.
PAGEBIN	The output bin number.
PAGEDUPLEX	Duplex control: • 0 = SIMPLEX • 1 = DUPLEX • 2 = DUPLEX TUMBLE

AFPIN filter settings

Use the filter chain editor to configure the AFPIN filter, or a settings file for default settings that can be used by multiple projects. See *AFPIN filter GUI reference* on page 18 and *AFPIN filter settings file* on page 17.

Note: Any configuration made in the filter chain editor overrides the corresponding configuration in the settings file.

AFP resources and search order

When you configure the AFPIN filter, you specify which AFP resources to use. Separate multiple paths using semicolons, and use wildcards to specify multiple files. Specify the paths in the order you want them to be searched.

You can also specify defaults that are used when there is no other way to determine, for example, the name and size of the TrueType font to use.

The AFP resources are searched in the following order:

1. In the inline resource group
2. In the configured paths
3. In the configured paths for the default resource groups and default resources

Note: External resource groups are only searched in the paths specified for resource groups.

AFPIN filter settings file

In the settings file, you specify default settings that can be used by multiple projects. Use the following syntax and one keyword per row when you enter the keywords:

```
<keyword>=<value>
```

For information about the keywords, see *AFPIN filter GUI reference* on page 18.

A sample settings file is included in the AFPIN filter installation. It is located in the following directory:

```
<installation_directory>/Common/modules/filters/AFPIN.settings
```

Example 19 *Contents in the AFPSettings.txt*

```
AFP_FONT_PATHS=C:/Font/;C:/AFPRes/*.300;D:/Res/X0*
AFP_IMAGE_PATHS=C:/Images/;C:/AFPRes/*.psg;D:/Res/S1*
AFP_EXTERNAL_FORMDEF=C:/AFPRes/F1FMDEF
AFP_IGNORE_INLINE_FORMDEF=yes
```

To create a settings file

1 Copy the sample file.

2 Enter keywords and their values.

3 Save the file and add it to a resource set connected to the Platform.

AFPIN filter GUI reference

The AFPIN filter converts AFPDS input to an internal XML format. The converted input is sent to a PreformatIN Event for processing.

You can use the keywords listed below in the settings file.

Settings	
Settings file name	The settings file to be used. Keyword: `AFP_SETTINGS_FILE`
Font paths	The AFP font resources to be used, for example: `C:/AFPRes/*.300;D:/Res/X0*;./External/Fonts` Keyword: `AFP_FONT_PATHS`
Image paths	The AFP image resources to be used, for example: `C:/Images/;C:/AFPRes/*.psg;./External/Images` Keyword: `AFP_IMAGE_PATHS`
Overlay paths	The AFP overlay resources to be used, for example: `./External/Overlays;C:/AFPRes/*.ove` Keyword: `AFP_OVERLAY_PATHS`
Resource group paths	The AFP external resource groups to be used, for example: `./External/ResGroup;C:/AFPRes/*.grp` Keyword: `AFP_RESGROUP_PATHS`
Default resources paths	The AFP default external resources to be used, for example: `./External;C:/Overlays/;C:/Fonts/;D:/Res/` Keyword: `AFP_DEFAULTRES_PATHS`
User GCGID map file name	The GCGID mapping file to be used. Keyword: `AFP_USER_GCGID_MAP_FILENAME`
GCGID generic masks	The GCGID generic masks to be used, for example: `U000%04X;X000%04X;UNIC%04X` Keyword: `AFP_GCGID_GENERIC_MASKS`

Settings	
Font mapping file name	The font mapping file to be used. Keyword: `AFP_FONT_MAPPING_FILENAME`
Default codepage name	The default code page to be used. If not specified here or in a settings file, `IBM CP 500` is used. Keyword: `AFP_DEFAULT_CODEPAGE_NAME`
Default font name	The default TrueType font to be used. If not specified here or in a settings file, `Courier New` is used. Keyword: `AFP_DEFAULT_FONT_NAME`
Default font size	The default font size in points to be used. If not specified here or in a settings file, `10` is used. Keyword: `AFP_DEFAULT_FONT_SIZE`
Default character	The default character to be used. If not specified here or in a settings file, the space character (Unicode X20) is used. Keyword: `AFP_DEFAULT_CHARACTER`
External formdef	The external AFP form definition to be used, for example: `C:/AFPRes/F1FMDEF` If you only specify the file name, and not the path, the paths specified for default external resources are searched. **Note:** The external form definition is only used when the **Ignore inline formdef** is set to Yes or form definition is not available in the inline resource group. Keyword: `AFP_EXTERNAL_FORMDEF`
Ignore inline formdef	Specifies if the inline AFP form definition in the document resource group should be ignored. **Yes** – The inline form definition is ignored. **No** – The inline form definition is used. **<Default>** – If not specified in a settings file, the inline form definition is used. Keyword: `AFP_IGNORE_INLINE_FORMDEF`

Settings	
Capture page TLE	Specifies if AFP TLEs on page level should be captured and converted to variables. See also *Capturing and reusing AFP objects* on page 16. **Yes** – The TLEs are captured. **No** – The TLEs are ignored. **<Default>** – If not specified in a settings file, the TLEs are captured. Keyword: `AFP_CAPTURE_PAGE_TLE`
Capture page group TLE	Specifies if AFP page group and document TLEs should be captured and converted to variables. See also *Capturing and reusing AFP objects* on page 16. **Yes** – The TLEs are captured. **No** – The TLEs are ignored. **<Default>** – If not specified in a settings file, the TLEs are captured. Keyword: `AFP_CAPTURE_PG_GROUP_TLE`
Capture BNG names	Specifies if BNG (Begin Named Group) names should be captured and converted to page property `AFP_BNG_NAME` available for re-use. See also *Capturing and reusing AFP objects* on page 16. **Yes** – The BNG names are captured. **No** – The BNG names are ignored. **<Default>** – If not specified in a settings file, the BNG names are ignored. Keyword: `AFP_CAPTURE_BNG_NAMES`

Settings	
NOP masks	Masks used for extracting NOP records. Use the following syntax: `[apf_level]:[starts_with]:<property_name>[;]` *apf_level* – The level where the NOP is extracted. If not specified, NOPs for all levels are extracted. Available levels are BGN and BPG. *starts_with* – Initial pattern of the NOP value to extract. If not specified, all available NOPs are extracted. *property_name* – Name of the variable that will contain the NOP. You can not use the following as variable names. The names are restricted to page description in the LXF file: • PAGEDUPLEX • PAGETRAY • PAGEBIN • PAGEORIENTATION • PAGEMEDIA • PAGEWIDTH • PAGEHEIGHT • MEDIUMMAP • MEDIAORIENT • AFP_BNG_NAME For more information and examples, see *Extracting NOP records* on page 14. Keyword: AFP_NOP_MASKS=[*apf_level*]:[*starts_with*]:<*property_name*>[;]
Use name referencing for missing overlays	Specifies how missing AFP overlays should be handled. **Yes** – If an overlay in the internal LXF format with the same name is available, it replaces the missing overlay. **No** – The missing overlay is ignored. **<Default>** – If not specified in a settings file, the missing overlay is ignored. Keyword: AFP_USE_NAME_REF_FOR_MISSING_OVERLAYS

Settings	
Use name referencing for missing images	Specifies how missing AFP images should be handled. **Yes** – If an image with the same name in PNG, TIFF or JPG format exists, it replaces the missing image. **No** – The missing image is ignored. **<Default>** – If not specified in a settings file, the missing overlay is ignored. Keyword: `AFP_USE_NAME_REF_FOR_MISSING_IMAGES`
Default double byte mapping	Only applicable when double-byte fonts are used in the AFP input data. The default double-byte code point to be used. If not specified here or in a settings file, `JAPANESE` is used. Keyword: `AFP_DEFAULT_DOUBLE_BYTE_MAPPING`
SDF binary output	The internal document format to be used. **Yes** – The binary LXF format SDF is used. **No** – The text based LXF format is used. **<Default>** – If not specified in a settings file, the text based LXF format is used. Keyword: `AFP_SDF_BINARY_OUTPUT`