Ricoh ProcessDirector

White Paper–Using the Enhance AFP Function

Enhance AFP

For information not in this manual, refer to the Help System in your product.

Version 3.3.0

Notices

Eighth edition (March 2015)

This edition applies to Ricoh ProcessDirector for AIX, Version 3 Release 3 (Program Number 5765-H30), to Ricoh ProcessDirector for Linux, Version 3 Release 3 (Program Number 5765-H30), to Ricoh ProcessDirector for Windows, Version 3 Release 3 (Program Number 5765-H30), and to all subsequent releases and modifications until otherwise indicated in new editions.

Internet

Visit our home page: `http://rpp.ricoh-usa.com/`

You can send comments by e-mail to **`printpublication@ricoh-usa.com`** or by mail to:

```
Ricoh Company, Ltd.
6300 Diagonal Hwy 004
Boulder, CO 80301-9270
U.S.A.
```

TABLE OF CONTENTS

About this white paper

Who should read this white paper 4
List of abbreviations 4
Related information 4

1 Enhance AFP

Introduction 7
Data type and enumeration definitions 8
Document update requests and their attributes 13
CoverBlock request 14
RemoveText request 15
LinearBarcode request 16
DataMatrixBarcode request 19
QR Barcode request 20
InsertText request 22
InsertTle request 24
FindText request 25
FindLinearBcocaBarcode request 28
FindTle request 31
Suppress request 32
Format strings 33

Notices

Trademarks 40

About this white paper

This white paper provides reference information about the Enhance AFP function.

Who should read this white paper

This white paper is for skilled administrators and programmers who want to configure and use Enhance AFP control files. Work with your Ricoh Production Print support representative as you do the tasks described in this white paper.

List of abbreviations

AFP™	Advanced Function Presentation™
AMB	Absolute Move Baseline
AMI	Absolute Move Inline
BCOCA™	Bar Code Object Content Architecture™
BNG	Begin named group
BPG	Begin page
ENG	End named group
EPG	End page
HRI	Human-readable information
IMB	Intelligent Mail bar code
IMM	Invoke medium map
IPDS™	Intelligent Printer Data Stream™
MO:DCA™	Mixed Object Document Content Architecture™
OMR	Optical mark recognition
PTOCA	Presentation Text Object Content Architecture
QR	Quick response
TLE	Tagged Logical Element
TRN	Transparent data

Related information

Visit the Ricoh Production Print Web site to see the most recent version of this white paper, and to find documentation for related products:

Ricoh Production Print Information Center (http://info.rpp.ricoh-usa.com/help/index.jsp)

1. Enhance AFP

- **Introduction**
- **Data type and enumeration definitions**
- **Document update requests and their attributes**
- **Format strings**

Steps based on certain step templates can use an Enhance AFP control file to change the content of AFP files. The step templates are BuildAFPFromDocuments, CreateAFPJobsFromDocuments, and, for the Document Pool extension, BuildEnhanceAFPFile. These steps call a program that is called *Enhance AFP* in this white paper. You can work with your support representative to create control files that contain rules for the changes.

These are examples of the changes you can make using Enhance AFP:

- Insert new bar codes or text.
- Remove obsolete content, such as bar codes, text, or OMR marks.
- Find variable text or bar code content to use as a basis for generating new content.
- Insert, find, or remove index tags (TLEs).

Messages and some of the instructions related to Enhance AFP refer to the function as *adf_extract*.

Example files for using Enhance AFP are located on the product CD in the `/samples` directory. After installation, you can find them in the directory `/`*`path`*`/extensions/doc/samples/`.

Visit the Ricoh Production Print Web site to see the most recent version of the documentation:

Ricoh Production Print Information Center (http://info.rpp.ricoh-usa.com/help/index.jsp)

Introduction

You use Enhance AFP to create "print" MO:DCA™ files from one or more "source" MO:DCA files. The print files are often units of work that are printed or inserted. The source files are usually files that are downloaded from an external source. Preprocessing, such as transforming to MO:DCA or indexing, is often required beforehand, using Enhance AFP.

Print files (production jobs) are produced from source files (original jobs) by assembling or "extracting" documents, or page groups, from the source files in a configurable manner. Enhance AFP includes various capabilities to update the documents, such as removing obsolete content or adding new content to them as they are extracted. The size (in sheets) of the generated print files and the order of the extracted documents are also configurable.

The source files to be processed by Enhance AFP must conform to these requirements:

- Source files must be MO:DCA.
- Source files must adhere to the AFP data standard described in the documentation for the Manufacturing Optimization feature and the Document Pool extension. For example:
  - Each document in an AFP file must be bounded by BNG and ENG structured fields.
  - All resources must be inline.
  - Each document must contain its own medium map.
  - The AFP file must conform to certain formatting rules so that text, bar code, and cover block enhancements can be added to its documents.

Limitations

Limitations of Enhance AFP include:

- Enhance AFP considers a sheet to be an "inserter sheet", which is the sheet size from the point of view of the inserter (usually 8.5" x 11"). This is not the printer sheet, which might consist of two inserter sheets (in the case of 2-up). When determining document sheet counts, we assume that "normal" page to partition placement occurs and that there is no enhanced n-up. That is, for simplex, it is assumed that each page occupies one inserter sheet, and for duplex, two pages occupy one sheet.
- Enhance AFP assumes that fonts on the original page are allocated in sequential manner; if you are going to add text to a page with a text block, additional local font IDs must still be available.
- All measurement units are in inches, except for some attributes that translate directly to BCOCA™ parameters (such as bar code height).
- Enhance AFP assumes a "natural" ordering of logical pages in the AFP it processes. This assumption is used to determine the sequence of pages and sheets, front versus back sheet placement in the duplex case, and left versus right placement in the 2-up case. Actual placement of pages onto the form by the form definition (n-up) or printer settings (cutsheet emulation) is independent from this assumption. Natural ordering means:
  - The first logical page in a mailpiece corresponds to the front of sheet sequence 1.
  - In the duplex case, logical pages that correspond to the front and back of a single sheet occur together, with the back page immediately following the front page.

Data type and enumeration definitions

Enhance AFP uses these types and enumerations for some attributes. The enumeration names should be spelled and capitalized exactly as shown for attribute values. When an optional attribute is not specified, a default is used that varies by type; the attribute descriptions specify which value is the default for each type. Depending on context, if an invalid value is specified for these data types, an error can result or the default value is assumed.

Alphanumeric

For this generic string type, any printable ASCII character is valid. An empty string value is assumed when an attribute of this type is not specified.

Color

Table 1. Enum values for Color

Enum value	Description
White	(Default) The print element should be the color of the medium.
Black	The print element should be black.

FormatString

This type is used to build dynamic content for document update requests. Document update requests that generate new content (such as insert text, linear bar code, Data Matrix bar code, and QR code bar code requests) use a simple formatting language by which a user can configure dynamic content. Format strings can consist of these elements:

- Literal data
- Pre-defined keywords
- Variables from the temporary variable pool
- Mailpiece (document) metadata
- Attributes
- Formatting information
- Functions
- Line separator

GenerationMethod

This type is used to specify the mechanism by which new MO:DCA™ content is generated. Not every value of this enumeration makes sense for specific applications, depending on context. For example, "BCOCA™" is not a valid value for describing an insert text request. The default value to use when this attribute is not specified also depends on context. Each of these methods should result in output identical in appearance and function, from a physical print point of view. Choice of method usually depends on printer compatibility, because not all IPDS™ printers support all methods (specifically, some newer bar codes types are not supported in BCOCA on older printers).

Table 2. Enum values for GenerationMethod

Enum value	Description
Generic	(Default) Indicates that Enhance AFP should use the method that guarantees compatibility on all IPDS printers.
BCOCA	Indicates that a BCOCA object should be generated to represent the content.
Font	Indicates that a PTOCA object with transparent text should be generated to represent the content. The referenced font resource must be available when printing.
DrawRule	Indicates that a PTOCA object using rules should be generated to represent the content.

HriLocation

This type is used to indicate the placement of the BCOCA-generated HRI for linear bar codes, if enabled.

Table 3. Enum values for HriLocation

Enum value	Description
Default	(Default) Indicates the BCOCA or printer default HRI location should be used.
Above	Indicates the HRI should be presented above the bar code symbol.
Below	Indicates the HRI should be presented below the bar code symbol.

Integer

This is an integer type (base 10). Only characters 0-9 are expected. A default value of 0 is assumed when an attribute of this type is not specified.

LinearBarcodeType

This enumerated type lists the linear bar code types that Enhance AFP supports. All linear bar code types defined in the BCOCA architecture are included.

Table 4. Enum values for LinearBarcodeType

Enum value	Description
None	(Default) No type specified; the bar code is not generated.
3of9	Generate a Code 39 (3-of-9 Code), AIM USS-39 bar code.
2of5_Interleaved	Generate an Interleaved 2-of-5, AIM USS-I 2/5 bar code.
POSTNET	Generate a POSTNET bar code (includes PLANET code bar codes).
MSI	Generate an MSI (modified Plessey code) bar code.
UPC_VersionA	Generate a UPC/CGPC Version A bar code.
UPC_VersionE	Generate a UPC/CGPC Version E bar code.
UPC_TwoDigitSupplemental	Generate a UPC - Two-digit Supplemental bar code.
UPC_FiveDigitSupplemental	Generate a UPC - Five-digit Supplemental bar code.
EAN8	Generate an EAN 8 (includes JAN-short) bar code.
EAN13	Generate an EAN 13 (includes JAN-standard) bar code.
2of5_Industrial	Generate an Industrial 2-of-5 bar code.
2of5_Matrix	Generate a Matrix 2-of-5 bar code.
Codabar	Generate a Codabar, 2-of-7, AIM USS-Codabar bar code.
Code128	Generate a Code 128, AIM USS-128 bar code.
EAN_TwoDigitSupplemental	Generate an EAN Two-digit Supplemental bar code.
EAN_FiveDigitSupplemental	Generate an EAN Five-digit Supplemental bar code.
RM4SCC	Generate a RM4SCC bar code.
JapanPostal	Generate a Japan Postal bar code.
AustralianPostal	Generate an Australian postal bar code.
Code93	Generate a Code 93 bar code.
IMB	Generate a USPS 4state - Intelligent Mail bar code.

Location

This type is expected on attributes that describe the location at which a print element should be placed on a page, relative to the logical page origin. The X and Y coordinates should be specified together separated by a comma; for example 1.25,8.0. Units are in inches. A default value of 0,0 is assumed when a location attribute is not specified.

Orientation

This type is expected on document update request attributes that describe how a generated print element should be oriented relative to the logical page origin.

Table 5. Enum values for Orientation

Enum value	Description
Degrees0	(Default) The print element should be oriented at 0 degrees.
Degrees90	The print element should be oriented at 90 degrees.
Degrees180	The print element should be oriented at 180 degrees.
Degrees270	The print element should be oriented at 270 degrees.

PlacementRule

This type is used to specify what pages in a document should be targeted for an update request, or similar action. These enumerated values describe various patterns of page sequences.

Table 6. Enum values for PlacementRule

Enum value	Description
None	(Default) No rule indicated; this has the effect of disabling the update request.
FirstFrontOnly	The update request should be applied only to the first front-facing page of each document.
FirstBackOnly	The update request should be applied only to the first back-facing page of each document. For a simplex job, this rule never applies.
AllFronts	The update request should be applied to all front-facing pages.
AllBacks	The update request should be applied to all back-facing pages. For a simplex job, this rule never applies.
AllPages	The update request should be applied to all pages.
LastPage	The update request should only apply to the last logical page of each document (front or back is not a factor).
AllFrontsLeftAndRight	The update request should only be applied to all front-facing pages and specifies that different locations should be used for left- and right-oriented pages, if 2-up is enabled.
SecondFrontOnly	The update request should be applied to the second front-facing page only. If the document has none (that is, the document contains a single sheet), this rule does not apply.

Enum value	Description
SecondBackOnly	The update request should be applied to the second back-facing page only. If the document has none (that is, the document contains a single sheet, or is simplex), this rule does not apply.
LastFrontOnly	The update request should be applied to the last front-facing page only.
LastBackOnly	The update request should be applied to the last back only. For a simplex job, this rule does not apply. Not all duplex documents contain a final logical page; if your last duplex page does not contain a final logical page, do not use this request.

Real

This is a real numeric type. Decimal fractions are allowed. Only characters 0-9 and "." are expected. A default value of 0.0 is assumed when an attribute of this type is not specified.

Rectangle

This type is expected on attributes that specify the location and size of a rectangular region. The X, Y coordinates and X, Y extent of the rectangle should be specified together separated by commas; for example 1.25,8.0,2.0,2.0. Units are in inches. A default value of 0,0,0,0 is assumed when a location attribute is not specified.

Regex

An attribute with a Regex type takes an extended regular expression. A specification for the regular expression support is available on The Open Group Web site (http://www.opengroup.org/onlinepubs/009695399/basedefs/xbd_chap09.html).

Trigger

A trigger is an expression, similar to format strings, that is evaluated in a boolean context. Triggers are normally used in Enhance AFP to implement conditional application of document update requests. Trigger expressions are fully evaluated to a string value, which is then interpreted as a boolean. If a trigger evaluates to the strings "0", "0.0", or to an empty string, the trigger is false. Any other string is considered to be true.

If a trigger attribute is expected but not given, the resulting behavior is the same as if a trigger that is always true were given (Enhance AFP internally assumes a trigger expression of '1'). This is not the same result as giving an empty string as a trigger expression, which is always false.

Trigger expressions support all features of format strings, but with these special features and exceptions:

- The given value is wrapped in @ symbols as a convenience. This is usually done because variable names and functions are the primary content of trigger expressions, and letting @ be omitted makes for more readable expressions. As a result, when using literal strings in trigger expressions, they must be wrapped in single quotation marks.
- When the function TRIM() is used in a trigger expression (for example, TRIM(column_name)), it is evaluated in the context of the current document.
- When the function EXISTS() is used in a trigger expression, it can be used to determine if a Find element was found in the current page.

- The operators == and != are supported. For example, the trigger expression APP_NAME== 'Statements' evaluates to true only when the variable APP_NAME is the string "Statements". The == and != comparisons are strictly string comparisons and not numeric ('0' is not the same as '00').

YesNo

Attributes that are of a boolean on/off nature take a value from the YesNo enumeration.

Table 7. Enum values for YesNo

Enum value	Description
No	(Default) Indicates the function or property is disabled or not set.
Yes	Indicates the function or property is enabled or set.

Document update requests and their attributes

This section describes the Enhance AFP document update requests and their attributes.

These are the Enhance AFP document requests:

- AfpCoverBlock
- AfpRemoveText
- AfpLinearBarcode
- AfpDataMatrixBarcode
- AfpQRBarcode
- AfpInsertText
- AfpInsertTle
- AfpFindText
- AfpFindLinearBcocaBarcode
- AfpFindTle
- AfpSuppress

Document update requests have associated attributes. Each attribute consists of a name, an index, and a value. The Enhance AFP™ control file must have one attribute per line. Lines beginning with a pound sign (#) are comments and are ignored. Use this format for each attribute line:

```
attribute_name[attribute_index]=attribute_value
```

The square brackets and equal sign (=) are required and are the basis for parsing the name, index, and value. White space is not used as a basis for parsing. The name, index, and value can be any alphanumeric character, including white space.

The attribute index is a label that you use to coordinate the update requests that work together. For example, one update request might define the data to be included in a bar code, another might define the type of bar code, and a third might define the x,y coordinates for placing it on the page. All the requests would share the same attribute index value; for example: InsertBarcode. As a rule, the value used for the attribute index is arbitrary. Some attributes work with each other, so their indexes must match. Other than that, the actual values do not matter. Usually, either arbitrary integers are used, or, for document update request attributes, it is helpful to use descriptive terms (such as InserterBarcode) as attribute indexes. The index values appear in the log and can be helpful for determining how configuration affects observed behavior.

If the Enhance AFP control file contains more than one instance of a given attribute, the last instance in the control file is used. The values taken by attributes might be numeric or alphanumeric, or they might be restricted to a defined list of names.

When you configure an update request, a *.Name attribute is not required but is recommended. A name is generated automatically if not specified, but since, in some cases, requests can refer to each other by name, specifying your own name lets you control this. The name used is arbitrary, but using a name descriptive for the update request is recommended because this name appears in log output. Also, this name is built into the MO:DCA as a structured field name or comment for update requests that generate new content. This lets you easily identify the changes that Enhance AFP made when you view the afpdmp output of generated print files.

One or more attributes of the form `Afp.<RequestType>.*` are required for each update request and are related by a common attribute index. Position in the Enhance AFP control file does not matter. If you use 8 attributes to set up a linear bar code, you must use the same attribute index on all 8 attributes, and you must not use this index on any other update request attribute.

Update requests also take an optional description attribute. If this is set, this string can be built into the generated MO:DCA for update requests that generate new content, as with the name.

Some of these attributes specify values and use units defined in the BCOCA architecture. For more information, see the *Bar Code Object Content Architecture™ Reference*. These attributes are listed as having a type of "BCOCA-defined". In most cases, when optional attributes of type "BCOCA-defined" are not given, Enhance AFP builds in values that indicate that BCOCA or printer defaults should be used.

Requests that start with the word "Find" (for example, "FindText") do not create new content or alter existing content, unlike other update requests. Instead, they locate existing content and let other requests suppress the found content, or use the content or metrics of the found content, for generating new content. When a find request locates a page, variables are inserted into a temporary variable pool and are available for any other request to use on that page using the format string language or as values of metric-oriented attributes. The variables exist only on the page where the find request found a match. Find request variables have the find request name as a prefix, followed by a period, which is followed by a name appropriate for the type.

Keep in mind that in the control file, you might see the Enhance AFP function referred to as *adf_extract*.

CoverBlock request

A CoverBlock request generates a solid rectangle. Use a white cover block to blank out an area on a page.

Table 8. Attributes that define a CoverBlock request

Attribute	Re-quired	Type	Description
Afp.CoverBlock.Color	Yes	Color	The color of the cover block.
Afp.CoverBlock.Description	No	Alphanumeric	A description for this request.

Attribute	Re-quired	Type	Description
Afp.CoverBlock.Name	No	Alphanumeric	The name of this request. If not used, a distinct name is generated automatically.
Afp.CoverBlockPlacementRule	No	PlacementRule	The placement rule to be used for deciding what pages should have the cover block applied.
Afp.CoverBlock.Rectangle	No	Rectangle	The location and extent of this cover block. The default value is 0,0,0,0.
Afp.CoverBlock.RightRectangle	No	Rectangle	The location and extent of this cover block on right-oriented pages if 2-up is enabled and the placement rule is AllFrontsLeftAndRight. If not specified, the standard location and extent are used for all pages.
Afp.CoverBlock.Trigger	No	Trigger	Enables conditional application of this request. If the trigger resolves to false for the scope in which the trigger value is evaluated (each page for which PlacementRule is true), this request is skipped for that scope.

RemoveText request

A RemoveText request removes text from an AFP™ document.

Table 9. Attributes that define a RemoveText request

Attribute	Re-quired	Type	Description
Afp.RemoveText.CharSetName Afp. RemoveText.CodePageName Afp. RemoveText.CodedFontName	Yes	Alphanumeric	Either a coded font or both a character set and code page are required. All instances of text using the given font are removed.
Afp.RemoveText.Description	No	Alphanumeric	Specifies a description for this request.
Afp.RemoveText.Name	No	Alphanumeric	The name of this request. If not specified, a distinct name is generated automatically.

Attribute	Re-quired	Type	Description
Afp.RemoveText.PlacementRule	No	PlacementRule	Specifies the placement rule to be used for deciding what pages should have the remove text request applied.
Afp.RemoveText.Trigger	No	Trigger	Enables conditional application of this request. If the trigger resolves to false for the scope in which the trigger value is evaluated (which for this request is each page for which PlacementRule is true), this request is skipped for that scope.

LinearBarcode request

A LinearBarcode request generates a linear bar code.

Some of these attributes specify values and use units defined in BCOCA™. For more information, see the BCOCA™ documentation. These attributes are listed as having a type of "BCOCA-defined". In most cases, when optional attributes of type "BCOCA-defined" are not given, Enhance AFP™ builds in values that indicate that BCOCA™ or printer defaults should be used.

Table 10. Attributes that define a LinearBarcode request

Attribute	Re-quired	Type	Description
Afp.LinearBarcode.Content	Yes	FormatString	Specifies the content of this bar code.
Afp.LinearBarcode.Context	No	Context	Specifies the context to be used for this bar code.
Afp.LinearBarcode.Description	No	Alphanumeric	Specifies a description for this request.
Afp.LinearBarcode.GenerationMethod	No	GenerationMethod	Specifies how the bar code should be represented in MO:DCA™ (BCOCA™ is used by default). If **Generic** is specified, the resulting output type depends on the type specified. Generic for IMB results in DrawRule, and Generic for all other Types specified uses BCOCA™.

Attribute	Re-quired	Type	Description
			Note Only IMB bar codes support types other than BCOCA™.
Afp.LinearBarcode.Height	No	BCOCA-defined	Specifies the height of this bar code. Any value greater than 0 and less than or equal to X'7FFF' is valid. A presentation device default is used if this attribute is not given, or is given with values 0 or X'FFFF', or with any invalid value.
Afp.LinearBarcode.IncludeHri	No	YesNo	Specifies whether the BCOCA-provided HRI should be generated for this bar code.
Afp.LinearBarcode.IncludeHriLocation	No	HriLocation	Specifies the location of the BCOCA-provided HRI, if enabled.
Afp.LinearBarcode.Like	No	Alphanumeric	Specifies a Find request whose results should be used for parameters of this request that are not explicitly specified.
Afp.LinearBarcode.Location	No	Location	Specifies the location of this bar code.
Afp.LinearBarcode.Modifier	Yes	BCOCA-defined	Specifies the modifier for this bar code.
Afp.LinearBarcode.ModuleWidth	No	BCOCA-defined	Specifies the module width of this bar code. Any value greater than 0 and less than or equal to X'FF' is valid. The value X'FF' is the BCOCA™ default indicator that means use the presentation device default. If an invalid value or no value is given, the default indicator is used. For IMB, the default X'FF' means print at the longer length. Any other value means print the IMB as short as possible given the implementation limitations.
Afp.LinearBarcode.Name	No	Alphanumeric	The name of this request. If not specified, a distinct name is generated automatically.

Attribute	Re-quired	Type	Description
Afp.LinearBarcode.Options	No	Integer	Specifies a bit field to enable defined bar code options. These options are Enhance AFP-specific, not BCOCA-specific. Only one option is supported (enabled with value 1), called SKIP_ON_BAD_CONTENT. Normally when a content format string resolves to an empty or invalid value for a specific document, Enhance AFP™ generates an error. When this option is set, however, no error is flagged; instead, no bar code is generated for the document.
Afp.LinearBarcode.Orientation	No	Orientation	Specifies the orientation of this bar code.
Afp.LinearBarcode.PlacementRule	No	PlacementRule	Specifies the placement rule to be used for deciding what pages should have the bar code applied.
Afp.LinearBarcode.RightLocation	No	Location	Specifies the location of this bar code on right-oriented pages if 2-up is enabled and the placement rule is AllFrontsLeftAndRight. If not specified, the standard location is used for all pages.
Afp.LinearBarcode.Trigger	No	Trigger	Enables conditional application of this request. If the trigger resolves to false for the scope in which the trigger value is evaluated (which for this request is each page for which PlacementRule is true), this request is skipped for that scope.
Afp.LinearBarcode.Type	Yes	LinearBarcode-Type	Specifies the type of this bar code.
Afp.LinearBarcode. WideToNarrowRatio	No	BCOCA-defined	Specifies the wide to narrow ratio (weNe) of this bar code. If not set or given with a value less than 1.0 or greater than 4.0, a presentation device default value is used.

DataMatrixBarcode request

A DataMatrix barcode request generates a DataMatrix barcode, either as a BCOCA object, a PTOCA object using draw rules, or as a font.

Some of these attributes specify values and use units defined in BCOCA™. For more information, see the BCOCA™ documentation. These attributes are listed as having a type of "BCOCA-defined". In most cases, when optional attributes of type "BCOCA-defined" are not given, Enhance AFP™ builds in values that indicate that BCOCA™ or printer defaults should be used.

Table 11. Attributes that define a DataMatrixBarcode request

Attribute	Re-quired	Type	Description
Afp.DataMatrixBarcode.Content	Yes	FormatString	Specifies the content of this bar code.
Afp.DataMatrixBarcode.Context	No	Context	Specifies the context to be used for this bar code.
Afp.DataMatrixBarcode.Description	No	Alphanumeric	Specifies a description for this request.
Afp.DataMatrixBarcode.ElementSize	No	BCOCA-defined	Specifies the size of the elements to use in the bar code.
Afp.DataMatrixBarcode.GenerationMethod	No	GenerationMe-thod	Specifies how the bar code should be represented in MO:DCA™ (BCOCA™ is used by default).
Afp.DataMatrixBarcode.Like	No	Alphanumeric	Specifies a Find request whose results should be used for parameters of this request that are not explicitly specified.
Afp.DataMatrixBarcode.Location	No	Location	Specifies the location of this bar code.
Afp.DataMatrixBarcode.Name	No	Alphanumeric	The name of this request. If not specified, a unique name is generated automatically.
Afp.DataMatrixBarcode.NumberOfRows	No	BCOCA-defined	Specifies the number of rows that the generated bar code should have. If not specified, the minimum number of rows needed to encode the specified content and achieve a square symbol is used. If specified, the row size must also be specified. If the specified number of rows is too small to encode the

Attribute	Re-quired	Type	Description
			specified data, additional rows are used.
Afp.DataMatrixBarcode.Orientation	No	Orientation	Specifies the orientation of this bar code.
Afp.DataMatrixBarcode.PlacementRule	No	PlacementRule	Specifies the placement rule to be used for deciding what pages should have the bar code applied.
Afp.DataMatrixBarcode.RightLocation	No	Location	Specifies the location of this bar code on right-oriented pages if 2-up is enabled and the placement rule is AllFrontsLeftAndRight. If not specified, the standard location is used for all pages.
Afp.DataMatrixBarcode.RowSize	No	BCOCA-defined	Specifies the number of elements that each row in the generated bar code should have. If not specified, the minimum number of elements needed to encode the specified content and achieve a square symbol is used. If specified, the number of rows must also be specified. If the specified row size is too small to encode the specified data, additional elements are used.
Afp.DataMatrixBarcode.Trigger	No	Trigger	Enables conditional application of this request. If the trigger resolves to false for the scope in which the trigger value is evaluated (each page for which PlacementRule is true), this request is skipped for that scope.

QR Barcode request

A QR barcode request generates a QR barcode as a BCOCA object.

Some of these attributes specify values and use units defined in BCOCA™. For more information, see the BCOCA™ documentation. These attributes are listed as having a type of "BCOCA-defined". In most cases, when optional attributes of type “BCOCA-defined” are not given, Enhance AFP™ builds in values that indicate that BCOCA™ or printer defaults should be used.

Table 12. Attributes that define a QRBarcode request

Attribute	Re-quired	Type	Description
Afp.QrBarcode.CodePageName	No	Alphanumeric	Specifies a target codepage for the content of this request. The input content is assumed to be UTF8, regardless of actual program execution locale.
Afp.QrBarcode.Content	Yes	Alphanumeric	Specifies the content of this bar code.
Afp.QrBarcode.Context	No	Integer	Specifies the context to be used for this bar code.
Afp.QrBarcode.ElementSize	No	Integer	Specifies the size of the elements to use in the bar code.
Afp.QrBarcode.ErrorCorrectionLevel	No	Alphanumeric	Specifies the error correction level of this barcode. Values must be one of "Level_L", "Level_M", "Level_Q", "Level_H". These represent error correction levels defined by the QR barcode standard. If no value is provided, Level L is used which corresponds to the minimum amount of error correction.
Afp.QrBarcode.Like	No	Alphanumeric	Specifies a Find request whose results should be used for parameters of this request that are not explicitly specified.

Attribute	Re-quired	Type	Description
Afp.QrBarcode.Orientation	No	Orientation	Specifies the orientation of this bar code.
Afp.QrBarcode.Version	No	Context	Specifies the version of this barcode. Value must be numeric, greater than or equal to 0 and less than or equal to 40. These represent versions (which corresponds to symbol size) defined by the QR barcode standard. If no value is provided, a level of 0 is used which results in the smallest possible symbol.

InsertText request

An InsertText request generates new PTOCA content.

Usually, the metric-oriented attributes on an InsertText request specify literal values such as **Degrees0**, **X0MYFONT**, or **1.0,1.0**. However, they can also refer to find request-generated variables in the temporary variable pool. Additionally, the Afp.InsertText.Like attribute can be specified to indicate that any metric for which an attribute is not given should be set to the metric value of the found object. This can be used as a shortcut to avoid having to specify each individual metric attribute.

Table 13. Attributes that define an InsertText request

Attribute	Re-quired	Type	Description
Afp.InsertText.CharacterSetName Afp.InsertText.CodePageName Afp.InsertText.CodedFontName	Yes	Alphanumeric	Either both a character set name and code name page or a coded font name must be provided to indicate the font to use for the InsertText request.
Afp.InsertText.Content	Yes	FormatString	Specifies the content of this InsertText request. Line separators are supported and result in an evenly spaced block of text with one printed line per logical line. A useful example is for address blocks. Format strings for InsertText request might include variable names referring to the content

Attribute	Re-quired	Type	Description
			found by a FindText request, such as `MyFind.Content`.
Afp.InsertText.Context	No	Context	Specifies the context to be used for this InsertText request.
Afp.InsertText.Description	No	Alphanumeric	Specifies a description for this request.
Afp.InsertText.Increment	No	Location	Specifies the X,Y distance, in inches, that should appear between lines of text when presenting a multi-line format string.
Afp.InsertText.Like	No	Alphanumeric	Specifies a Find request whose results should be used for parameters of this request that are not explicitly specified.
Afp.InsertText.Location	No	Location	Specifies the location of this InsertText request.
Afp.InsertText.Name	No	Alphanumeric	The name of this request. If not specified, a distinct name is generated automatically.
Afp.InsertText.Orientation	No	Orientation	Specifies the orientation of this InsertText request.
Afp.InsertText.PlacementRule	No	PlacementRule	Specifies the placement rule to be used for deciding what pages should have the InsertText request applied.
Afp.InsertText.RightLocation	No	Location	Specifies the location of this InsertText request on right-oriented pages if 2-up is enabled and the placement rule is AllFrontsLeftAndRight. If not specified, the dimensions the standard location is used for all pages.
Afp.InsertText.Trigger	No	Trigger	Enables conditional application of this request. If the trigger resolves to false for the scope in which the trigger value is evaluated (which for this request is each page for which PlacementRule is true), this request is skipped for that scope.

InsertTle request

An InsertTle request generates new document-level TLEs.

Created TLEs are inserted immediately after the BNG of the target document. The content of created TLEs uses format strings and might refer to the results of any find request. When using predefined Enhance AFP™ keywords in a content string, page-level keywords are not supported (for example, **cur_page_in_mp**). Results are undefined if unsupported keywords are used.

Table 14. Attributes that define an InsertTle request

Attribute	Re-quired	Type	Description
Afp.InsertTle.Content	No	FormatString	Specifies the content of the TLE, in ASCII. This value is translated to EBCDIC when the TLE is created. This attribute is required if Afp.InsertTle.Like is not given.
Afp.InsertTle.Description	No	Alphanumeric	Specifies a description for this request.
Afp.InsertTle.Like	No	Alphanumeric	Specifies a Find request whose results should be used for parameters of this request that are not explicitly specified.
Afp.InsertTle.Name	No	Alphanumeric	The name of this request. If not specified, a distinct name is generated automatically.
Afp.InsertTle.TleName	No	Alphanumeric	Specifies the actual TLE name to be built into the X'02' triplet, in ASCII. This value is translated to EBCDIC when the TLE is created. This attribute is required if Afp.InsertTle.Like is not given, or if it is given but does not refer to a FindTle request from which a TleName can be determined.
Afp.InsertTle.Trigger	No	Trigger	Enables conditional application of this request. If the trigger resolves to false for the scope in which the trigger value is evaluated (which for this request type is each extracted document), this request is skipped for that scope.

FindText request

A FindText request searches for existing PTOCA content and makes the content and metrics of the found text available to other requests.

When a FindText request finds matching content, it creates temporary variables and makes them available to other update requests.

Table 15. Variables that the FindText request creates

Variable	Description
FindName.CharacterSetName	If the found text uses a font based on a character set and code page, this variable has the character set name that the found text uses.
FindName.CodePageName	If the found text uses a font based on a character set and code page, this variable has the code page name that the found text uses.
FindName.CodedFontName	If the found text uses a font based on a coded font, this variable has the coded font name that the found text uses.
FindName.Content	The actual content, in ASCII, of the found text.
FindName.Location	The location at which the text was found.
FindName.Orientation	The orientation of the found text.

You can search for text using any combination of these criteria:

Area

Limit the search to a specific rectangular region on any page.

Content pattern

Search for text that matches a given extended regular expression.

Font

Specify either a coded font or both a character set and a code page to search for text using this font.

Orientation

Search for text with a specific orientation.

Each criterion is optional. If no criteria are given, the FindText request matches any text. By defining search criteria, you can narrow the scope of the search until only the desired text is targeted.

If an area is given to limit the search, only text whose position can be parsed are candidates to be found. For Enhance AFP to understand the position, the text must be positioned with absolute positioning controls AMI and AMB structured fields. If positioning is done with relative moves, or multiple TRN structured fields occur after a single absolute position is set, text might be ineligible for finding based on area.

The actual text found and made available to other request types in the *FindName*.Content temporary variable can include a configurable number of TRNs. The first TRN that matches the given criteria starts the found text. It ends when the configured maximum match count is met, or when a TRN is processed that does not match the criteria. If TRN-matching stops because of divergence from the search criteria but the limit is not reached, additional text later in the page (potentially in other PTXs) that matches the criteria becomes part of the found text, until the limit is reached.

FindText requests can include an option that causes an error if the maximum find limit is exceeded. For example, this option could be set in conjunction with setting the limit to 1. This causes an error if two or more instances of matching text are found, which ensures that unintentional text is never found (and perhaps altered) unintentionally.

Table 16. Attributes that define a FindText request

Attribute	Re-quired	Type	Description
AfpFindText.Area	No	Rectangle	Specifies the area in which text should be found. This is measured such that the down and right directions are the positive numbers. The origin of the text must fall in this area; if other parts of the text extend beyond the area, it is still a match. Text that does not originate in this area is ignored. If not given, text at any position is eligible to be found.
AfpFindText.CharacterSetName AfpFindText.CodePageName AfpFindText.CodedFontName	No	Alphanumeric	Either both a character set name and a code page name or a coded font name must be provided to indicate the font for the text to be found. Text using other fonts is ignored. If not given, text using any font can found.
AfpFindText.Description	No	Alphanumeric	Specifies a description for this request.
AfpFindText.EnforceHardLimit	No	YesNo	When **Yes**, an error is issued when the limit given by Afp. FindText.Limit is exceeded. Default behavior is to ignore additional matching instances without issuing an error.
AfpFindText.Limit	No	Numeric	Defines a maximum number of expected matching TRNs. The default value is 1. TRNs past this limit are not included in the found text returned from a FindText request. Behavior when the limit is exceeded

Attribute	Re-quired	Type	Description
			depends on the value of the Afp.FindText.EnforceHardLimit attribute. If multiple TRNs are found, their content is concatenated to produce the complete matching text for a given page. The value 0 can be used to indicate that no limit should be used, so any and all matching text on a page is accumulated and returned.
AfpFindText.Name	Yes	Alphanumeric	The name of this find request. Results of this find request can be referred to using this name. In practice, an explicit, distinct name should always be given. Otherwise, if multiple find requests exist with the same name, the first item found with the common name is the found object.
AfpFindText.Orientation	No	Orientation	Specifies the orientation criteria for finding text. Any text with a different orientation is ignored. If not given, text at any orientation can be found.
AfpFindText.Pattern	No	Regex	Specifies the pattern-matching criteria for finding text. Text that does not match is ignored. If not given, text with any content can be found.
AfpFindText.PlacementRule	No	Alphanumeric	Limits the scope of the FindText request to specific pages in each document. If not given, all pages in a document are searched for matching text.

Attribute	Re-quired	Type	Description
AfpFindText.RightArea	No	Rectangle	If set, this area is used to limit the find on right-oriented pages. If not set, the area specified by Afp.FindText.Area is used for all pages.
AfpFindText.Trigger	No	Target	Enables conditional application of this request. If the trigger resolves to false for the scope in which the trigger value is evaluated (which for this request is each page for which PlacementRule is true), this request is skipped for that scope.

FindLinearBcocaBarcode request

A FindLinearBcocaBarcode request searches for existing BCOCA™ linear bar codes and makes the content and metrics of the found bar code available to other requests.

Note

As the name implies, this request works only on BCOCA™ bar codes of linear types. For example, it cannot locate Data Matrix bar codes or POSTNET bar codes built as text using a bar code font. When this request finds matching content, it creates temporary variables and makes them available to other update requests. Many of these variable are useful only for creating new bar codes, but some (such as content, orientation, and location) can be used for creating any new object.

Table 17. Variables that the FindLinearBcocaBarcode request creates

Variable	Description
FindName.Content	The actual content, in ASCII, of the found bar code.
FindName.IncludeHri	This variable is set to **Yes** if the found bar code uses human-readable information.
FindName.Height	The height of the found bar code.
FindName.HriLocation	The HRI location of the found bar code.
FindName.Location	The location at which the bar code was found.
FindName.Modifier	The modifier of the found bar code.
FindName.ModuleWidth	The module width of the found bar code.
FindName.Orientation	The orientation of the found bar code.
FindName.Type	The type of the found bar code (for example, 3of9, POSTNET).
FindName.WideToNarrowRatio	The weNe ratio of the found bar code.

You can search for bar codes using any combination of these criteria:

Area

Limit the search to a specific rectangular region on any page.

Content pattern

Search for bar codes that match a given extended regular expression.

Height

Search for bar codes with a specific height.

Modifier

Search for bar codes with a specific modifier, as defined in BCOCA™.

Orientation

Search for bar codes with a specific orientation.

Type

Search for bar codes of a specific type; for example, 3of9, POSTNET, or Code128.

Each criterion is optional. If no criteria are given, the FindLinearBcocaBarcode request matches any linear BCOCA™ bar code. By defining search criteria, you can narrow the scope of the search until only the desired bar code is targeted.

If the search criteria result in multiple matching bar codes on a page, the first matching bar code encountered is returned.

Table 18. Attributes that define a FindLinearBcocaBarcode request

Attribute	Re-quired	Type	Description
Afp.FindLinearBcocaBarcode.Area	No	Rectangle	Specifies the area in which bar codes should be found. The origin of the bar code must fall in this area; if other parts of the bar code extend beyond the area, it is still a match. Bar codes that do not originate in this area are ignored. If not given, bar codes at any position can be found.
Afp.FindLinearBcocaBarcode. Description	No	Alphanumeric	Specifies a description for this request.
Afp.FindLinearBcocaBarcode.Height	No	BCOCA-defined	Specifies the bar code height to find. If not specified, bar codes with any height can be found.
Afp.FindLinearBcocaBarcode.Modifier	No	BCOCA-defined	Specifies the bar code modifier to find. If not specified, any bar code modifier of the given type can be found.

Attribute	Re-quired	Type	Description
Afp.FindLinearBcocaBarcode.Name	Yes	Alphanumeric	The name of this find request. Results of this find request can be referred to using this name. In practice, an explicit, distinct name should always be given. Otherwise, if multiple find requests exist with the same name, the first item found with the common name is the found object.
Afp.FindLinearBcocaBarcode.Orientation	No	Orientation	Specifies the orientation criteria for finding bar codes. Any bar code with a different orientation is ignored. If not given, bar codes at any orientation can be found.
Afp.FindLinearBcocaBarcode.Pattern	No	Regex	Specifies the pattern-matching criteria for bar code content. Bar codes whose content does not match are ignored. If not given, a bar code with any content can be found.
Afp.FindLinearBcocaBarcode.PlacementRule	No	PlacementRule	Limits the scope of the find request. If not given, all pages are searched for a matching bar code.
Afp.FindLinearBcocaBarcode.RightArea	No	Rectangle	If set, this area is used to limit the find on right-oriented pages. If not set, the area specified by Afp.FindLinearBcocaBarcode.Rectangle is used for all pages.
Afp.FindLinearBcocaBarcode.Trigger	No	Trigger	Enables conditional application of this request. If the trigger resolves to false for the scope in which the trigger value is evaluated (which for this request is each page for which PlacementRule is true), this request is skipped for that scope.
Afp.FindLinearBcocaBarcode.Type	No	LinearBarcode-Type	Specifies the type of linear BCOCA™ bar code to find. If not specified, bar codes of any type can be found.

FindTle request

A FindTle request searches for existing Tagged Logical Elements (TLEs) in an extracted document, at any scope (document/mailpiece level or page level).

When a FindTle request finds matching content, it creates temporary variables and makes them available to other update requests.

Table 19. Variables created by the FindTle request

Variable	Description
FindName.Content	The content of the TLE as encoded in the X'36' triplet. Found TLE content is assumed to be EBCDIC and is translated to ASCII when published.
FindName.TleName	The name of the TLE as encoded in the X'02' triplet. Found TLE names are assumed to be EBCDIC and are translated to ASCII when published.

You can search for TLEs by TLE name. If you do not specify a TLE name, the FindTle request matches any TLE.

If the search returns multiple matching TLEs for a mailpiece, the TLE name and content of each matching TLE are published as variables to the format string variable pool. The variables are inserted or updated when each subsequent TLE is found.

FindTLE results are cleared at the end of each document.

Table 20. Attributes that define a FindTLE request

Attribute	Re-quired	Type	Description
Afp.FindTle.Description	No	Alphanumeric	Specifies a description of this request.
Afp.FindTle.Name	No	Alphanumeric	The name of this find request. Results of this find request can be referred to using this name. In practice an explicit name should always be given, and should be distinct. Otherwise, if multiple find requests exist with the same name, then the first item found with the common name is the found object.
Afp.FindTle.Suppress	No	YesNo	If this value is **Yes**, the found TLE is changed to a NOP. If suppression is enabled and multiple TLEs are found in a mailpiece, all are suppressed.

Attribute	Re-quired	Type	Description
Afp.FindTle.TleName	No	Alphanumeric	The name of the TLE to be found. The name should be given in ASCII and is translated to EBCDIC for comparison to TLE names found in the extracted AFP™ stream. Names must match exactly; comparisons are case sensitive. If not given, TLEs with any name can be found.
Afp.FindTle.Trigger	No	Trigger	If the trigger resolves to false for the scope in which the trigger value is evaluated (which is every time a TLE is encountered in the extracted AFP™ stream), then this request is skipped for that scope. If it evaluates to true, then the request is applied. If no trigger attribute is given, behavior is the same as if a trigger is present and always true.

Suppress request

A Suppress request can be used with Find requests that target page-level printable objects (for example, FindText and FindLinearBcocaBarcode) to cause the found content not to be printed.

A Suppress request should refer to a defined Find request by name.

A found object is suppressed on a given page when:

1. The referenced Find request is executed on the page.
2. The referenced Find request finds matching content on the page.
3. The Suppress request is executed on the page.

Using a Suppress request with a Find request allows for a finer level of control than using a Remove request. For example, you might want to find some content on every page of a document, but keep it from printing on only the first page. You can do this by setting up a Find request that runs on every page and a Suppress request that runs only on the first page of each document. The found content is available to generate new content on any page, while only specific instances are suppressed.

Table 21. Attributes that define a Suppress request

Attribute	Re-quired	Type	Description
Afp.Suppress.Description	No	Alphanumeric	Specifies a description for this request.
Afp.Suppress.Name	No	Alphanumeric	The name of this request. If not specified, a distinct name is generated automatically.
Afp.Suppress.NameToSuppress	Yes	Alphanumeric	The name of a Find request that corresponds to the printable object to be suppressed. If the referenced Find request does not find an object in the scope in which this Suppress request is applied, the Suppress request has no effect.
Afp.Suppress.PlacementRule	No	PlacementRule	Limits the scope of the Suppress request. If not given, all pages are targeted.
Afp.Suppress.Trigger	No	Trigger	Enables conditional application of this request. If the trigger resolves to false for the scope in which the trigger value is evaluated (which is page scope for Suppress requests), this request is skipped for that scope.

Format strings

Document update requests that generate new content (such as insert text, linear bar code, Data Matrix bar code, and QR code bar code requests) use a simple formatting language by which a user can configure dynamic content.

Format strings can consist of these elements:

- Literal data
- Pre-defined keywords
- Variables from the temporary variable pool
- Mailpiece (document) metadata
- Attributes
- Formatting information
- Functions
- Line separator

Literal data

This information is presented *as is*. Anything that is quoted (using single quotation marks) or is not another element type is considered to be literal data.

Pre-defined keywords

Pre-defined keywords are pre-defined names that represent values that Enhance AFP calculates internally. These are indicated by including a defined keyword name in @ symbols. Keyword names are not case-sensitive. These keywords are available to use in format strings for building new content, such as bar codes and text.

Keyword validity and value resetting operate according to a simple scope hierarchy. There are four levels to the hierarchy. Each keyword is "native" to one level of the hierarchy. Keywords can be used in their native hierarchy or in lower hierarchies, but not in higher levels; otherwise, errors or unexpected results can occur. Keyword values are invariant in a given instance of their native hierarchy, but normally change at boundaries. This list contains the four hierarchy levels (the highest level is listed first):

Table 22. Enum values for pre-defined keywords

Hierarchy	Description
All print jobs	This level applies to a higher level than the extracted AFP stream, but applies instead to properties that span all output print files.
Current print job	This level applies when processing of the extracted AFP stream is outside of any document (BNG/ENG group); for example, when resource groups or document-level TLEs are being managed.
Mailpiece (document)	This level applies when processing of the extracted AFP stream is inside a BNG/ ENG group but is outside any BPG/EPG; for example, when IMMs or document-level TLEs are being managed.
Page	This level applies when processing of the extracted AFP stream is inside a logical page (a BPG/EPG group); for example, when bar codes or text are being created.

Enhance AFP supports these pre-defined keywords:

Table 23. Supported pre-defined keywords

Keyword	Hierarchy	Description
total_mp_in_pj	Print job	The total number of documents in the current print job.
total_sheets_in_pj	Print job	The total number of sheets in the current print job.
total_pages_in_mp	Mailpiece (document)	The total count of logical pages in the current document.
total_sheets_in_mp	Mailpiece (document)	The total count of sheets in the current document.
cur_mp_in_extract	Mailpiece (document)	The sequence of the current document in the overall run, starting at 1. The value returned for the last document is always equal to the total document count.

Keyword	Hierarchy	Description
cur_mp_in_pj	Mailpiece (document)	The sequence of the current document in its containing print job, starting at 1.
cur_page_in_mp	Page	The sequence of the current logical page in its containing document, starting at 1.
cur_sheet_in_mp	Page	The sequence of the current sheet in its containing document, starting at 1.
cur_page_in_pj	Page	The sequence of the current logical page in its containing print job, starting at 1.
cur_sheet_in_pj	Page	The sequence of the current sheet in its containing print job, starting at 1.

Almost all format strings are evaluated at a page level, so usually all keywords listed in the table above are valid to use. This is true for most update requests; for example, creating bar codes or text. However, in some cases format strings are evaluated at levels higher than Page, so be careful not to use keywords in lower hierarchies. If you do so, results might be unexpected.

For example, {cur_page_in_mp}} does not work at the mailpiece (document) level, because there is no "current page" until processing of logical pages begins.

As an example of non-Page level format string evaluation, consider the TLE create request. When used to create TLEs at the document level, the format string given by the Afp.InsertTle.Content attribute is evaluated at the document level. In this case, keywords native to the Page hierarchy level cannot be used, but any keyword in document or higher can be used.

Variables from the temporary variable pool

Enhance AFP has a temporary variable pool that grows and shrinks according to business rules and AFP processing scoping rules. Variables in this pool can be referred to by name by any format string, and they resolve to the corresponding variable value.

The variable pool is populated only with FindText requests. FindText variables exist only for the page in which the request found a match. The variables are cleared at the end of each page.

Attributes

Any attribute from the Enhance AFP control file can be included in a content string. The attribute name must be delimited with @ symbols and the attribute index must be given after a period at the end of the attribute name, not in square brackets. You can list any attribute, not only recognized ones listed in this white paper that drive insert text requests. You can have, for example, "MyAttr[0]=XYZ". If you put @MyAttr.0@ in a format string as part of a bar code's format, it resolves to "XYZ". In other words, you can introduce variables into format strings that can change from job to job this way.

Formatting information

You can use format specifiers to determine how a keyword, function result, or literal value should appear. You might want values to appear in a fixed width format, with leading zeros or spaces. The format string architecture lets you do this using C format specifiers.

Format information should be placed between % symbols, just after the initial @ symbol that begins a keyword or function. The value between the % symbols should be a C format specifier, such as %d, or % s. Minimum and maximum field width specifiers can also be used, such as %6s, or %08d. These format specifiers are used as is, so any valid C-format specifier can be used. However, the types must match– that is, %d must be used with integer data, otherwise an error occurs.

For example, @%02d%cur_sheet_in_mp@ indicates that the cur_sheet_in_mp value should be displayed using two characters, and adds leading zeros if necessary.

Functions

You can use functions to add flexibility to how dynamic content is included. A function must be enclosed in @ symbols, just like keywords, and consists of a name, parenthesis, and an argument list. Formatting information can be used for functions.

Each function has its own argument list definition. Each function can take some control arguments, and each one takes at least one value argument, which are the values to be manipulated for inclusion in the resulting string. A value argument can be a literal, enclosed in single quotation marks, or a keyword. When a keyword is used as a value argument, it is resolved for the current page before being passed to the function.

The functions that Enhance AFP supports are:

- BIN(value, trueChar)

  The BIN() function takes a string as the value parameter, interprets each character as a boolean flag, and returns the resulting integer value.

  Value can be a literal or a keyword. Each character in the string is evaluated as a boolean based on the trueChar parameter. If a character in the string matches trueChar, it is considered true, otherwise it is considered false. The resulting binary number (right-most bit is least significant) is converted to a decimal integer and is returned as the result.

  For example, BIN('YYNNY', 'Y') results in 25 (which is binary 11001 in decimal).

  This function can be useful for generating inserter control information, in a bar code or HRI, based on a string that represents bins on an inserter. In the above example, YYNNY could be document metadata that specifies that bins 1, 4, and 5 are to be pulled for this document on the inserter.

- EXPR(expression)

  The EXPR() function lets an arbitrary arithmetic expression be evaluated. The result is the integer value to which the expression evaluates. The expression can contain one or more integer literals or keywords, along with the operators +, -, *, /, and %. Unary minus is not supported. Operator precedence is the set {*, /, %} evaluated left to right, followed by {+, -} evaluated left to right.

- SUBSTR(value, start, len)

  The SUBSTR() function takes a keyword or literal and returns the specified substring, starting at start, for the length len. Start is 0-based. If len is longer than the available string length, the entire length of the string after start is returned without errors. If value is empty, or start points after the available length of value, the function resolves to an empty string. Start can be negative, in which case the substring starts that many characters from the end of the string.

- SUM(value1, value2, ..., value n)

  The SUM() function takes one or more keywords or literal integer values and returns the arithmetic sum of all the arguments. It can be more efficient the EXPR(), so use this if doing simple sums. A typical use of this function can be to change a pre-defined keyword to be zero based; for example, SUM('-1',CUR_MP_IN_PJ) returns the current document sequence starting at 0.

- TR(value, fromChars, toChars)

  The TR() function operates on value, translating characters listed in fromChars to characters in toChars. Characters between fromChars and toChars are linked positionally. For example,. TR (customer_name, 'abc', ' ') translates any instance of a, b, or c in customer_name to a space. The fromChars and toChars should be the same length. It is also possible for fromChars and toChars to specify other replacement keywords, document metadata, or attributes.

- TRIM(value)

  The TRIM() function removes any whitespace at the beginning of the string value. Embedded whitespace is not affected.

- EXISTS(value)

  The EXISTS() function returns the value 1 if the value exists with any value and returns 0 if the value does not exist as a valid variable. This function can be used with a FindText or other Find command to determine if the Find command actually found any data.

Line separator

The characters \n can be used as a line separator to logically separate lines of text for cases where multi-line text is supported. This is not a newline character but the literal backslash and "n" characters. As an example, this format string could be given to an insert text request to print an address block:

```
@address1@\n@address2@\n@address3@\n@address4@\n@address5@\n@address6@\n
```

Notices

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

```
Ricoh Company, Ltd.
6300 Diagonal Hwy 004
Boulder, CO 80301-9270
U.S.A.
```

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by Ricoh under terms of the Ricoh Customer Agreement, Ricoh Software License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-Ricoh products was obtained from the suppliers of those products, their published announcements or other publicly available sources. Ricoh has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-Ricoh products. Questions on the capabilities of non-Ricoh products should be addressed to the suppliers of those products.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Ricoh and Ricoh ProcessDirector are not affiliated with NEC Corporation.

Trademarks

These terms are trademarks or registered trademarks of Ricoh Co., Ltd., in the United States, other countries, or both:

- Advanced Function Presentation
- AFP
- Bar Code Object Content Architecture
- BCOCA
- InfoPrint
- Infoprint
- Intelligent Printer Data Stream
- IPDS
- Mixed Object Document Content Architecture
- MO:DCA
- Ricoh

Ricoh ProcessDirector White Paper–Using the Enhance AFP Function

EN US EN GB EN AU

3/2015