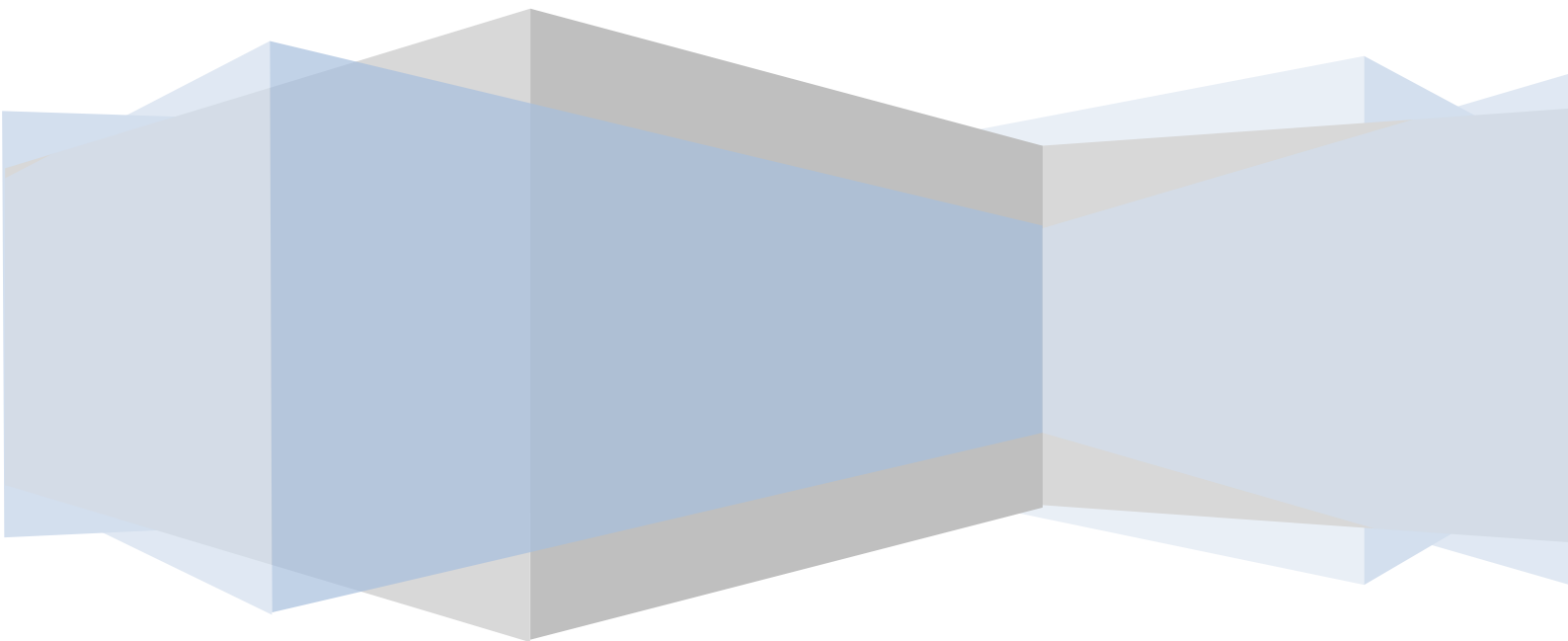




Proven Product, Proven Company

Function & Syntax

User Guide – Version 1.0

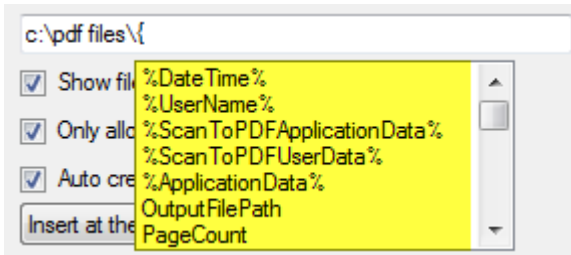


INTRODUCTION

ScanToPDF code products (and the plug-ins) have advanced functionality and syntax that may be used in some circumstances for added flexibility. This guide explains the syntax of each function. Each section details the syntax and functions available for each module of the software ; the core ScanToPDF products (Basic and Standard), and each of the plug-ins (where applicable)

SCANTOPDF FUNCTIONS & SYNTAX

All syntax and functions are used within curly braces {}. When a curly brace is used in one of the settings fields of ScanToPDF a small drop down list with the available syntax/functions will appear (as highlighted below)



To use the syntax you can scroll to it using the keyboard arrow keys and press return, or you can scroll using the mouse and then click the required item to populate the field settings with the selected item

Function Syntax	Description	Example Usage
%DateTime%:Date/Time format	Dates and times	Filenaming - C:\PDF\{%DateTime%:dd-MM-yyyy}
%UserName%	The name of the current logged on user	Filenaming - C:\PDF\{%UserName%\}{%increment%.pdf
%ScanToPDFApplicationData%	Provides the path of the ScanToPDF Application Data folder	
%ScanToPDFUserData%	Provides the path of the ScanToPDF User Data folder	
%ApplicationData%	The path to the ScanToPDF Application Data folder	
OutputFilePath	The path to the PDF file created	Used in Data Exporter for example to log the path to the created PDF file
PageCount	The number of pages in the PDF file created	Used in Data Exporter to log the number of pages in the created PDF file
ImportFilePath	The path of a file imported into ScanToPDF using the import from folder option	
BatchDocumentSavedCount	Number of document saved in a batch of documents	
BatchTotalSavedPageSavedCount	Number of pages saved in a batch of documents	
BatchCapturedPageCount	Number of pages captured in a batch of saved documents	
Date(<i>string</i>)	Return a date from a given string	{Date(strDate):ddMMyyy} - useful if the date (strDate) is part of a barcode or retrieved from OCR for example
Trim(<i>string</i>)	Removes all leading and trailing white-space characters from a string	Trim(" Hello") Result="Hello"
Capitalise(<i>string</i>)	Capitalise a string	
RemoveSpaces(<i>string</i>)	Removes ALL white-space characters from a string	RemoveSpaces("George Best") Result = "GeorgeBest"

RemoveChars(<i>string, characters</i>)	Remove a character (or list of characters) from a string	{RemoveChars(input,"e")} Input="Hello" Result="Hllo"
SubString(<i>string, startpos, length</i>)	Return a sub string of length length from within a string starting at position zero-based startpos	{SubString(%barcode%,0,3)} Barcode = 12345 Result = 123
GetFileInfo(<i>string</i>)	Return a FileInfo object for a given string	{GetFileInfo(ImportFilePath) CreationTime:yyyyMMdd} ImportFilePath=C:\pdf files Result="20110611"
GetFolderInfo(<i>string</i>)	Return a FolderInfo object for a given string	
Extract(<i>string, delimiter, field number</i>)	Extract a field from a string separated by field separator delimiter	{Extract(%barcode%,"-",2) %barcode%=BN-12345-99 Result = 12345
Ternary(<i>condition, expression1, expression2</i>)	If condition = true then resolve expression1 otherwise resolve expression2	{Ternary("Prefix=NBTI", {String.Format("C:\PDF\Invoices\{0}-{1}.pdf", ClientID, Number)}, {String.Format("C:\PDF\Checks\{0}-{1}.pdf", ClientID, Number)})} Prefix=NBTI ClientID=12345 Number=1 Result=C:\PDF\Invoices\12345-1.pdf Prefix=ABCD ClientID=12345 Number=1 Result=C:\PDF\Checks\12345-1.pdf
GetFileName(<i>string</i>)	Get the name of a file (including extension) for a provided string	{GetFileName(OutputFilePath)} OutputFilePath="c:\pdf files\12345.pdf" Result="12345.pdf"
GetFileNameWithoutExtension(<i>string</i>)	Get the name of a file without the extension for a provided	{GetFileNameWithoutExtension(OutputFilePath)} OutputFilePath="c:\pdf files\12345.pdf" Result="12345"
GetExtension(<i>string</i>)	Get the extension of a file for a provided string	{GetExtension(OutputFilePath)} OutputFilePath="c:\pdf files\12345.pdf" Result=".pdf"
GetFolderName(<i>string</i>)	Get the folder name of a provided string	{GetFolderName(OutputFilePath)} OutputFilePath="c:\pdf files\12345.pdf" Result="c:\pdf files"
Translate(<i>string, oldvalue, newvalue</i>)	Translate old value in a given string into new value (Both old value and new value can be multiple values separated by a Pipe Symbol)	{Translate(%barcode%,"SO IN","SALES ORDER INVOICE")} %barcode%=SO Result="SALES ORDER" %barcode%=IN Result="INVOICE"
Replace(<i>string, oldvalue, newvalue</i>)	Replace an old value in a given string with a new value	{Replace(%barcode%,"SO","SALES ORDER")} %barcode%="SO-12345" Result="SALES ORDER-12345"
MultiReplace()	Same as above but supports multi values in a list separated by (Pipe symbol)	{MultiReplace(%barcode%,"SO IN","SALES ORDER INVOICE")} %barcode%="SO-12345" Result="SALES ORDER-12345" %barcode%="IN-12345"

		Result="INVOICE-12345"
RegexReplace(<i>string, regular expression, new value</i>)	Replace a portion of a string that matches regular expression with new value	{RegexReplace(ImportFolder,"^C:\\George", "c:\\pdf files")}{%barcode%}.pdf ImportFolder=C:\\George Result=C:\\pdf files
ValueMatchesRegex(<i>string, regular expression</i>)	Returns a True or False if variable string matches regular expression	{ValueMatchesRegex(ED,"[0-9]{3}")} ED=123 Result=True

FILENAMER FUNCTIONS & SYNTAX

{%increment%}	Returns an incrementing number	For example used in filename C:\\pdf\\{%increment%:0000}.pdf Would produce C:\\pdf\\00001.pdf
{%version%}	Returns a unique version number for a file, increments until the filename used does not exist	For example used in filename C:\\pdf\\{%increment%}-{%version%}.pdf

BARCODE RECOGNITION FUNCTIONS & SYNTAX

{%barcode%}	Get a barcode using barcode recognition	C:\\pdf\\{%barcode%}.pdf
{GetBarcode()}	Get a barcode token using barcode recognition	{GetBarcode().Type.ToString()} would get the barcode type

BATCH SEPARATION FUNCTIONS & SYNTAX

{DocumentNumber}	The number of the document in the current batch	In batch scanning, each PDF file will have a unique document number in the batch, the document number starts at 1 (the first document in a batch), see below
{DocumentPageNumber}	The number of the page within the current document number	See example from batch visualization below
{DocumentIncludedPageCount}	The number of pages within the current document	See example from batch visualization below

