Kofax FraudOne

Service Program Configuration

Version: 4.5.0

Date: 2020-11-04

KOFAX

Table of Contents

Preface

The FraudOne service programs permit input and output of signature-relevant information directly into or out of the database. In order to get the best performance from the FraudOne Applications, it is recommended, that you never run more than one service program on one PC. If more Getter/Putter/SRF etc. instances are needed, you have to install them on separate PCs.

In standard usage, these programs can be employed as follows:

1. AccountLoader
2. ImageLoader
3. DataViewer
4. SignatureReferenceFilter
5. FraudFeedbackFileLoader
6. XML-Loader
7. Getter
8. Putter
9. DFP
10. ResultLoader
11. ResultWriter
12. PasswordEncoder
13. TableAccess

Related documentation

The full documentation set for Kofax FraudOne is available at the following location:

https://docshield.kofax.com/Portal/Products/FO/4.5.0-th2k87ey6r/FO.htm

In addition to this guide, the documentation set includes the following items:

Guides

- *Kofax FraudOne Administrator's Guide*
- *Kofax FraudOne Data Warehouse Installation and Operation Guide*
- *Kofax FraudOne Extended Reporting Features and Statistics*
- *Kofax FraudOne Feature Codes*
- *Kofax FraudOne Installation and Migration Guide*

- *Kofax FraudOne Java Client Customization Guide*
- *Kofax FraudOne Java Client Customization Layer*
- *Kofax FraudOne License Management*
- *Kofax FraudOne Report Component Installation Guide*
- *Kofax FraudOne SignCheck Result Codes*
- *Kofax FraudOne Standard Reporting Features and Statistics*
- *Kofax FraudOne The Book on CRS*
- *Kofax FraudOne Thin Client Customization Guide*
- *Kofax FraudOne Thin Client Customization Layer*

Interfaces

- *Kofax FraudOne Archive Interface Server*
- *Kofax FraudOne ASV Blackbox*
- *Kofax FraudOne Global Fraud Signature Web Service Developer's Guide*
- *Kofax FraudOne Common API Specifications for GIA Engines*
- *Kofax FraudOne Service Program Interfaces*
- *Kofax FraudOne User Login Procedure*
- *Kofax FraudOne Standard Teller Interface*
- *Kofax FraudOne Variant Cleanup Utility*

Online Help

- *Kofax FraudOne Administration Client Help*
- *Kofax FraudOne Java Client Help*
- *Kofax FraudOne Server Monitor Help*
- *Kofax FraudOne Thin Client Help*

Training

Kofax offers both classroom and computer-based training that will help you make the most of your Kofax FraudOne solution. Visit the Kofax website at www.kofax.com for complete details about the available training options and schedules.

Getting help with Kofax products

The Kofax Knowledge Base repository contains articles that are updated on a regular basis to keep you informed about Kofax products. We encourage you to use the Knowledge Base to obtain answers to your product questions.

To access the Kofax Knowledge Base, go to the Kofax website and select **Support** on the home page.

Note The Kofax Knowledge Base is optimized for use with Google Chrome, Mozilla Firefox or Microsoft Edge.

The Kofax Knowledge Base provides:

- Powerful search capabilities to help you quickly locate the information you need.

  Type your search terms or phrase into the **Search** box, and then click the search icon.
- Product information, configuration details and documentation, including release news.

  Scroll through the Kofax Knowledge Base home page to locate a product family. Then click a product family name to view a list of related articles. Please note that some product families require a valid Kofax Portal login to view related articles.
- Access to the Kofax Customer Portal (for eligible customers).

  Click the **Customer Support** link at the top of the page, and then click **Log in to the Customer Portal**.
- Access to the Kofax Partner Portal (for eligible partners).

  Click the **Partner Support** link at the top of the page, and then click **Log in to the Partner Portal**.
- Access to Kofax support commitments, lifecycle policies, electronic fulfillment details, and self-service tools.

  Scroll to the **General Support** section, click **Support Details**, and then select the appropriate tab.

Chapter 1

Prerequisites for the service programs

The prerequisites for the service programs are:

- All service programs need a Java runtime environment.
- The SignatureReferenceFilter and XML-Loader need additionally a TCP/IP connection to a SignBase server.
- The ResultLoader and ResultWriter need additionally a TCP/IP connection to a SignCheck server.
- The SignatureReferenceFilter and XML-Loader, ResultLoader, ResultWriter need additionally a TCP/IP connection to a SignBase server.
- AccountLoader, ImageLoader, FraudFeedbackFileLoader, Getter, Putter, DFP and TableAccess need a JDBC connection to the SignBase/SignCheck database and its associated JDBC classes.
- AccountLoader, Getter, SignatureReferenceFilter need a Sival license, if a mono signature cleaning takes place.
- All programs need a SignPlus2 license, if encrypted passwords are to be used.

Chapter 2

Configuration of the service programs

General configuration

The main configuration file of each service program has the name of the service program and the extension ".properties", e.g. the program Getter has the main configuration file Getter.properties. When different main configuration files for a service program are needed, it is possible to change the name by giving the program an argument.

Syntax

```
java –cp … de.softpro.signplus.service.<program> [conf-file [key1=value1
[key2=value2]]] …
```

where:

`program`

is the name of the service program

`conf-file`

is the name of the main configuration file

`key1`

is the first key in the main configuration file, whose value will be replaced by value1

`key2`

is the second key in the main configuration file, whose value will be replaced by value2 etc.

Example

Normally the Getter is started with the command

```
java –cp …. de.softpro.signplus.service.Getter
```

The name of the main configuration file is in this case Getter.properties.

But if the Getter is started with the command

```
java –cp …. de.softpro.signplus.service.Getter G1
```

then the name of the main configuration file is G1.properties.

All service programs have the following common configurations in the main configuration file.

Configuration of the main configuration file

Keys, that are marked with (f) can be defined with a formula:

```
${[<name>]<delm>[<function1>][<delm><function2>…]… }
```

where:

```
<name>
```

is a key from a hashtable

```
<delm>
```

is one of the characters "|", "?" or ":"

```
<functionx>
```

is one of the defined functions, analog to the formulas in the table properties files

See chapter Formulas.

Example

```
deleteDataFile =${BNO|TEST$*<=305?FMT0:FMT1}
```

This means: if the value of key BNO is less than or equals 305, then deleteDataFile is set to 0, otherwise it is set to 1.

Key	Default	Description
Settings on the GUI		
startLabel	"Start"	Label of the start-button
stopLabel	"Stop"	Label of the stop-button
exitLabel	"Exit"	Label of the exit-button
browseLabel	"Browse…"	Label of the browse buttons
dataLabel	"Data Directory"	Label of the input field for the data directory
logLabel	"Log File"	Label of the input field for the protocol file
title(f)	classname of the service program	Title of the window
license	"(unlicensed)"	Label of the licensee
windowWidth	screenwidth / 2	Width of the window in pixel
windowHeight	screenheight / 2	Height of the window in pixel
bankSpecificReleaseText	"bank-specific Package: "	Label for the release text
release	<unknown>	release text

Key	Default	Description

logFont	Windows setting	The font of the log.
		Syntax
		`„name,style,size“`
		where:
		`name`
		The name of the font.
		`style`
		The style of the font:
		0 - normal
		1 - bold
		2 - italic
		3 - bold and italic
		`size`
		The size of the font in pixels.
		Example
		`„Helvetica,0,16“`

logSelectedForeground	Windows setting	Color of the selected text in the log Syntax `red,green,blue` where each value is scalable from 0 (0%) to 255 (100%).
logSelectedBackground	Windows setting	Color of the selected background in the log
logForeground	Windows setting	Foreground color for messages without tracelevel in the log
logForeground1	Windows setting	Foreground color for messages with tracelevel ERROR in the log
logForeground2	Windows setting	Foreground color for messages with tracelevel WARNING in the log
logForeground4	Windows setting	Foreground color for messages with tracelevel DEBUG in the log
logForeground8	Windows setting	Foreground color for messages with tracelevel INFO in the log
logForeground16	Windows setting	Foreground color for messages with tracelevel RESOURCE in the log
logForeground32	Windows setting	Foreground color for messages with tracelevel SQL in the log
logForeground64	Windows setting	Foreground color for messages with tracelevel SUBSTITUTE in the log
logForeground128	Windows setting	Foreground color for messages with tracelevel SELECTION in the log
logForeground256	Windows setting	Foreground color for messages with tracelevel PERFORMANCE in the log

Key	Default	Description
logBackground	Windows setting	Color of the background in the log
logMaximumLines	10000	The maximal number of lines in the log area. Is this number reached, then as many lines beginning from the top are removed that logMinimumLines lines remain. This setting has no influence on the log written to a file. The log in a file is always complete.
logMinimumLines	1000	The minimal number of lines in the log area
displayBatchView	false	If set to true, the BatchView Panel is shown initial, instead of the log area.
helpFile	empty	If not empty, the name of the help file.
Default settings on the GUI		
logFile(f)	empty	Name of the protocol file
dataDir	<the current directory>	Directory of the data files
startNow	nodataSuffix	If set to “yes”, the program starts immediately processing of the data files after being activated. Otherwise the user must press the start button to begin the processing of the data files. **Note** If startNow=yes, saveLog=yes and wait=0, then the program is finished after processing all files.
saveLog	no	Shall the log be written to the log file? (yes/no)
traceLevel	0 (no trace)	Trace Level. The Trace-Level is the sum of the single levels (see Trace Levels), i.e. 9 is ERROR (1) and INFO (8): 1+8=9.
Definition of more resource files		
menuResource		Name of the resource file describing the menu bar.
Database settings		
ignoreSQLCode	empty	A comma-separated list of SQL return codes that can be ignored, i.e. processing will continue with 0 rows affected
dataErrorSQLCode	empty	A comma-separated list of SQL return codes indicating data errors
tryAgainSQLCode	empty	Comma-separated list of SQL-Error codes, that allow a re-execute of the failed SQL-Command, e.g. deadlock or timeout.
tryAgainWait	10	Time in seconds, before the SQL-Command is re-executed after an SQL-Error in the tryAgainSQLCode list.

Key	Default	Description
maxRetries	3	Maximum number of repetitions of an SQL-Command in the tryAgainSQLCode list
waitForConnection	10	Time in seconds to wait before re-trying to establish a connection to the database
commitCount	1	Number of data file lines to process, before an SQL COMMIT is executed. **Note** If reject file processing is enabled (rejectSuffix is not empty), the value for commitCount is set to 1
codepageDatabase	UTF-8	Name of the codepage of the database according to the IANA Charset Registry
specialRegisters	0	Number of variables to query from the database, these queries are normally table-independent
specialRegisterName1		Name of the 1st variable
specialRegisterName2		Name of the 2nd variable and so on
specialRegisterValue1		SQL-Command to query the value for the 1st variable
specialRegisterValue2		SQL-Command to query the value for the 2nd variable and so on
specialRegisterUpdateMode1	1	Determines the time for the query of variable 1: 1 - after reading one line from the data file 2 - after executing of all SQL-Commands for one line 4 - before executing of the SQL-Command for every single table 8 - after executing of the SQL-Command for every single table 16 - only once immediately after the start of the program Combinations can be achieved by adding the values.
specialRegisterUpdateMode2	1	Determines the time for the query of variable 2 and so on.
Password.Encryption	true	Enable password encryption of passwords in properties files.
Data file settings		
codepageDatafile	Cp858	Name of the codepage according to the IANA Charset Registry, in which the data in the data file are coded

Key	Default	Description
workSuffix	No working file	Suffix for working files. The data file is locked by creating an empty working file in the same directory, with the same filename, but with the extension workSuffix. This allows more than one program to run on the same data directory without processing the same file twice.
dataSuffix(f)	".dat"	Suffix of all data files
renameSuffix(f)	no Default	Suffix for renaming the data file. When specified, after processing the data file's dataSuffix is replaced by renameSuffix
renamePath	The data directory	Path for the data files that are to be renamed.
renameMaxRetries	3	Maximum number of retries when the rename of a data file fails, before an error is thrown.
renameWait	100ms	Time in ms to wait between two rename tries.
rejectSuffix(f)	empty	If rejectSuffix is not empty, then all records that could not be properly processed are written to a reject file. This file has the name of the data file and the extension „.rejectSuffix“.
deleteDataFile(f)	true	If set to true, the successfully processed data file is not renamed, but deleted. If an activation file exists, it is deleted.
errorSuffix(f)	".err"	Suffix for the error file. When the processing of the data file fails, the log is written to the file with the name of the data file and this extension.
workDirectory	empty	Sets the directory for the working files to a different directory than the data directory. Is necessary for the case that the data directory is a CD-ROM.
recordLength	0 (every line ends with CR/LF)	Record length of the data files. All records must have the same length. A record length of 0 indicates variable record lengths: all records end with CR/LF.
randomDataFileSelection	false	Shall the input files be processed in a random order? "true" makes sense, if many programs work on the same directory.
minFileAge	0s	The minimum age of a data file, possible unis are: s - second m - minute h - hour d - day w - week A value of 0 means: the age of the data file is ignored.

Key	Default	Description
maxFileAge	0s	The maximum age of a data file, possible unis are s - second m - minute h - hour d - day w - week A value of 0 means: the age of the data file is ignored.
dataFileIsZip	false	If true, then the data file is a zip file, every entry in this file is considered to be a record.
dataFileIsXML	false	If true, then the data file is an XML file. Which part of the file is a record depends of the used XMLReader class. If both dataFileIsXML and dataFileIsZip are false then the data file is a text file.
XMLSchema	defaultSchema	The schema for the used XML file. If empty then the schema from inside the XML file is used. This is defined with key "schemaLocation" in the root node, e.g. <?xml version="1.0" encoding="UTF-8"?><batch schemaLocation="serviceSchema5.xsd"> … If no schema is specified inside the XML file then defaultSchema.xsd is used. Only active if dataFileIsXML=true
XMLReader	de.softpro.XMLReader	The implementation of the XML reader. This must be an extension of de.softpro.XMLReader. If empty then the implementation from inside the XML file is used. This is defined with key "implementation" in the root node, e.g. <?xml version="1.0" encoding="UTF-8"?><batch implementation= "de.softpro.signplus.service.XMLReader"> … If no implementation is specified inside the XML file then de.softpro.XMLReader is used. Only active if dataFileIsXML=true
Database tables settings		
tables	0	Number of tables to process
tableResource1		Name of the 1st properties file describing the 1st table
tableResource2		Name of the 2nd properties file describing the 2nd table and so on

Key	Default	Description
newFunctions	0	Number of additional functions for the substitution process in the table properties
newFunction1		First additional function, consisting of: • the classname of the new function • the name of the new function • optional a list of arguments for the Constructor of the function classname, name and argument list are separated by „:“
newFunction1Delimiter	“:”	Alternative Delimiter for the 1st function’s argument list
newFunction2		2nd additional function and so on
newFunction2Delimiter	“:”	Alternative Delimiter for the 2nd function’s argument list
maxLines	0	The maximum number of data file lines, that the program is supposed to process. Is this number reached, the program will terminate. Currently activated only for the AccountLoader. maxLines=0: there is no maximum number of lines
reconnectLines	0	Number of data file lines to process during one connection. Is this number reached, the connection to the database is closed and then re-established and the program continues. reconnectLines=0: the connection persists
Program control		
wait(f)	0 (no wait, but STOP after processing all data files)	Number of seconds to wait after processing all data files before searching for new data files
waitForConnection	10	Number of seconds to wait if the connect to the database failed before retrying
projectStartClass	empty	The name of a class that is additional started when the start button has been pressed. If this class must run until the stop button is pressed, it should create its own thread. It is expected that this class has a constructor with an instance of de.softpro.signplus.service.Service as only parameter.
variablePrefix	empty	An additional prefix character for variables. Normally variables start with “${“. This character is appended to the prefix. If variablePrefix=X, then all variables must start with “${X”.
resourceComponent	service	Name of the component in MessageAA00 to get properties from the configuration server instead of a resource file.

Key	Default	Description
Report		
report	false	Is a report to be generated
reportOnlyDB	true	If true, only those records of a data file are reported that have changed the database; otherwise all records of a data file are reported.
reportDelimiter	„;“	Delimiting character for the fields of a report line
reportResource		Name of the properties file describing the format of the report and the path of the report file.
REPORT.PN	01 or the instance from the server manager (only if the program was started by the server manager)	The process number, can be used for the report file name

Configuration of the menu bar

The configuration of the menu bar takes place in a properties file, where all menus and menu items and their types are defined. The link to the program is made by an action name for each menu item.

Key	Default	Description
menubar	no default	A list of menus delimited by a blank. Each menu must be defined in this file with the menu as key. Example `menubar=menu1 menu2 menu3 …`
<menu1>	no default	A list of menu items delimited by a blank for the first menu. Each menu must be defined in this file with the menu as key. The menu item "–" has a special meaning: this is a separator. Each menu item in this list is the first part of further keys that describe properties of one menu item.
<menu2>	no default	A list of menu items delimited by a blank for the second menu and so on.
<menu>Scroll	false	true - the menu can be scrolled
<menu>ScrollMaxRows	20	The maximum number of visible menu items when scrolled
<menuitem>Label	no default	The name of this menu item
<menuitem>Action	no default	The action string of this menu item. This is the link to the calling program. The program has only to know the value of this key to work with this menu item.

Key	Default	Description
<menuitem>Accelerator	no Accelerator	The key combination which invokes the menu item's action listeners without navigating the menu hierarchy. It consists of `[<modifier>+[<modifier>+[<modifier>+[<modifier>+]]]]<keyname>` where modifier can be: • Shift • Ctrl • Meta • Alt and keyname is a normal printable character or one of the following: • F1, F2, F3, F4, F5, F6, F7, F8, F9, F10, F11, F12 • Tab • Escape • Insert, Delete, Home, End, PageUp, PageDown, Left, Right, Up, Down • Enter • Space • Num0, Num1 ,…, Num9 • Ctrl
<menuitem>Mnemonic	no Mnemonic	A character specifying the mnemonic value (Alt-<character>) to activate this menu item.
<menuitem>Icon	no Icon	Name of an icon for the menu item. Must be in the classpath.
<menuitem>Tooltip	no Tooltip	Tooltip text for the menu item.

Batch view

The programs AccountLoader, ImageLoader, SignatureReferenceFilter, FraudFeedbackFileLoader, Getter and ResultLoader have in common, that they process files in a directory. To control the state of processing of the data files you can activate the batch view on the GUI. This can be configured in the menu bar of the program. The action string ViewProgressAction is the link to the program. Normally this menu item is placed in the view Menu.

If the batch view is enabled, the display of the log is disabled automatically.

The batch view contains the following items:

- Selecting and deselecting the listing of data files in the 4 states ready, in progress, finished and rejected using checkboxes
- Activating bank-specific filters for the listing of data files
- 2 buttons for re-input of the selected reject files or all reject files respectively

- a refresh button for the list of data files
- the list of data files in the current data directory, filtered by the settings described above

The list of data files will be updated automatically every 5 seconds. Those data files with state in progress are shown with a progress bar in the status column displaying the progress of processing. This works also for programs that run on other PCs in the network, as long as they use the same data directory.

The Batch View can be configured with the following keys in the main configuration file:

Key	Default	Description
batchViewShow	false	If set to true, the Batch View will be displayed after starting the program, otherwise the log will be displayed.
batchViewTitle	"Batch Status"	Title of the Batch View dialog
batchViewFileTitle	"File Type"	Title of the Batch View file-selection
batchViewFile1	"Ready"	Name of the checkbox for selecting of the files with state "Ready for processing". This name is also used for the status in the status column of the list.
batchViewFile2	"In progress"	Name of the checkbox for selecting of the files with state "In progress".
batchViewFile3	"Finished"	Name of the checkbox for selecting of the files with state "Finished".
batchViewFile4	"Rejected"	Name of the checkbox for selecting of the files with state "Rejected". This means, the file has been processed, but at least one record has been rejected.
batchViewSelect1	false	Presetting for selection of files with state "Ready"
batchViewSelect2	false	Presetting for selection of files with state "In progress"
batchViewSelect3	false	Presetting for selection of files with state "Finished"
batchViewSelect4	false	Presetting for selection of files with state "Rejected"
batchViewFilters	0	Number of filters for the data file selection
batchViewFilterTitle	"File Filter"	Title of the Batch View filter-dialog
batchViewFilter1Label	"Filter 1"	Name of the 1st filter
batchViewFilter2Label	"Filter 2"	Name of the 2nd filter etc.

Key	Default	Description
batchViewFilter1Type	"char"	Type of the 1st filter. There are the following types: char - text int - a number range - a range from one to another number date - a date time - a time
batchViewFilter2Type	"char"	Type of the 2nd filter etc.
batchViewFilter1Digits	"1.1"	Area of the data file name, where the filter 1 is intended for. The first number is the start column, the second number is the number of digits.
batchViewFilter2Digits	"1.1"	Area of the data file name, where the filter 1 is intended for etc.
batchViewFilter1Select	false	Presetting for the activation of the 1st filter
batchViewFilter2Select	false	Presetting for the activation of the 2nd filter etc.
batchViewFilter1Value		Presetting for the value of the 1st filter. If this filter has the type "range", this key can contain 2 values, divided by comma.
batchViewFilter2Value		Presetting for the value of the 2nd filter etc.
batchViewFilterRangeLabel	"to"	Label for the 2nd value in case the filter type is "range"
batchViewButtonAllLabel	"Re-input all"	Label of the button for re-inputting all reject files
batchViewButtonSelectedLabel	"Re-input selected"	Label of the button for re-inputting the selected reject files
		Design of the Batch View list
batchViewListFont	"Helvetica,0,20"	The font to be used for the list
batchViewListColumnSelection	all columns are shown	A comma-separated list of column numbers that are supposed to be displayed
batchViewListColumns	0	Number of list columns
batchViewListTitle	"Files in directory"	Title for the list of data files
batchViewListTitle1		Heading for column 1
batchViewListTitle2		Heading for column 2 etc.

Key	Default	Description
batchViewListWidth1	Width of the window / number of columns	Width of column 1 in pixel
batchViewListWidth2	Width of the window / number of columns	Width of column 2 in pixel etc.
batchViewListAlignment1	RIGHT	Alignment for column 1. Possible values are: - RIGHT - CENTER - LEFT
batchViewListAlignment2	RIGHT	Alignment for column 2 etc.
batchViewListAttr1		Type of column 1. Is needed for sorting and modifying the display. Possible values are: Int - the column contains integer values Date - the column contains integer values, that can be interpreted as date in milliseconds. The display is according to the current date format Progress - the column contains either numbers between 0 and 1000, that are shown as progress bar, or ordinary text, that is displayed unchanged String - the column contains text
batchViewListAttr2		Type of column 1 etc.
batchViewListTitleFGColor		Foreground color of the title, as RGB-list, e.g. "255,0,0" is red
batchViewListTitleBGColor		Background color of the title
batchViewListHeaderFGColor		Foreground color of the heading
batchViewListHeaderBGColor		Background color of the heading
batchViewListRowFGColor		Foreground color of the unselected rows
batchViewListRowBGColor		Background color of the unselected rows
batchViewListSelectedRowFGColor		Foreground color of the selected rows
batchViewListSelectedRowBGColor		Background color of the selected rows
batchViewListRowHeight		The height of the rows, in pixel. If this height is too small to display the selected font entirely, the font's height is taken instead.

Key	Default	Description
batchViewListSelection	SINGLE	Sets the table's selection mode to: SINGLE - allows only single selections INTERVAL - allows a single contiguous interval MULTIPLE - allows multiple intervals

Chaining of properties files

There are some special keys that allow to take over keys from another properties file or a whole properties file in the current properties file.

Key	Description
$#USEresource=new_resource	Use resource file new_resource instead of resource file resource. This applies for all subsequent definitions of resource file resource.
$#INCLUDEppp_resource	Includes all keys from the properties file resource. ppp is the priority of this statement. ppp can be omitted, in this case the priority is assumed to be 0. ppp is a positive number and 0 is the lowest priority. resource is taken as the name of a properties file and its keys are added to this resource bundle if these keys do not exist there. But if two include files contain the same key with the same priority, a RuntimeException is thrown. If they have different priorities, the key with the lower priority is ignored.
$#INSERTppp resource:prefix	Insert parts of another properties file. resource is the properties file where all keys starting with 'prefix.' are added to the keys of this properties file regardless if they already exist or not, i.e. the keys that are added with this statement have precedence over the existing keys in the current properties file. ppp is the priority of this statement
$#UNDEF_key	Removes the key.
$#KEYppp resource:regex	Includes all keys from the properties file resource matching the regular expression regex. ppp is the priority of this statement. ppp can be omitted, in this case the priority is assumed to be 0. resource is taken as the name of a properties file.

Take care that you use every key only once. The key is delimited from its value by whitespace, “=” or “:”. If you use a key twice, one of them is completely ignored.

Example

```
$#INCLUDE file-a

$#INCLUDE file-b
```

One of these files will be ignored completely, without error message. To avoid this, the priority (as a part of the key) can be used:

```
$#INCLUDE1 file-a

$#INCLUDE2 file-b
```

Precedence of the key definitions:

1. The $#USE statement
2. keys from a $#KEY statement with high priority
3. keys from a $#KEY statement with low priority
4. keys from a $#INSERT statement with high priority
5. keys from a $#INSERT statement with low priority
6. keys in the properties file
7. keys from a $#INCLUDE statement with high priority
8. keys from a $#INCLUDE statement with low priority

Configuration of table properties files

Each table properties file describes the access to a single table of a database. Currently the databases db2, MSSQL-Server and oracle are supported.

Key	Default	Description
driver	"sun.jdbc.odbc.JdbcOdbcDriver"	Fully qualified class name of the driver.
URL		The database URL
catalog	empty	Catalog name of the database, not necessary for DB2
schema	empty	Schema of the database
name		The name of the table

Key	Default	Description

SP.name	empty	Name of a stored procedure. If not empty, then this stored procedure will be executed instead of performing a statement on this table. The input parameters for the stored procedure are the values of the first table columns, i.e. default1 contains the first parm, default2 the second etc. If there are more default… keys than the stored procedure parameters has, the remainder plays no role for the stored procedure.
		If the result of the stored procedure is a ResultSet, its columns are assigned to those columns if this table, that have an "S" in the type. The results are stored under the names of these columns in the hashtable.
		Example
		A stored procedure has 1 input parm and returns a ResultSet with 1 column and 3 rows. The following keys are defined:
		`SP.name=pGetCol`
		`name1=INPUT`
		`default1=<inputvalue>`
		`name2=S.SPRESULT`
		After execution of the stored procedure the following keys are defined:
		`SPRESULT.0=3`
		`SPRESULT.1=<row1>`
		`SPRESULT.2=<row2>`
		`SPRESULT.3=<row3>`

SP.catalog	value of catalog	Catalog name of the stored procedure
SP.schema	value of schema	Schema of the stored procedure
props		The number of connection properties. Only those properties are used whose name propKey… exists.
propKey1(f)		The name of the first property etc.
propValue1(f)		The value of the first property etc.
user		The user id, can be empty. It overwrites the user property key.
password		The password, can be empty. It overwrites the password property key.
incorrectPasswordCode	empty	A comma-separated list of all SQL return codes indicating that the userid/password combination was incorrect. If one of these return codes is returned when a connection was tried to establish, then a Userid/password dialog is shown for getting a valid combination.

Key	Default	Description
timeout	0 (no timeout)	The timeout in seconds for all SQL execute commands.
transactionIsolation	2	Isolation Level of database transactions: 1 - read uncommitted 2 - read committed 4 - repeatable read 8 - serializable
updateStatistics		The appropriate SQL command for updating the statistics of a table. Each ${1}-substring inside the command is replaced by the real table name.
autoCommit	no	If set to yes, then all its SQL statements will be executed and committed as individual transactions.
autoUpdate	X	Perform an UPDATE if INSERT returns the errorcode duplicateKeyCode: I - update only INSERT columns U - update only UPDATE columns N - simulate a successful INSERT other - an error message will occur
autoInsert	X	Perform an INSERT if UPDATE returns the errorcode notFoundCode: I - insert only INSERT columns U - insert only UPDATE columns other - do NOT perform autoInsert Example `AutoInsert=U`
duplicateKeyCode	-803	The return code of an INSERT Statement when the primary key already exists
notFoundCode	100	The return code of an UPDATE Statement when the specified row does not exist
useVariablesForWhere	true	If true, use a “?” as value for columns in the where-clause, the real values are bound to the column. If false, use literal values.
SELECT.Extension	empty	An extension of the SELECT command, e.g. a setting of SELECT.Extension=DISTINCT would lead to the command “SELECT DISTINCT…”
DELETE.Extension	empty	An extension of the DELETE command
INSERT.Extension	empty	An extension of the INSERT command
UPDATE.Extension	empty	An extension of the UPDATE command

Key	Default	Description
columns		The maximum number of columns to access. Only those columns are taken into account whose key name… is defined. This can be less than the number of all columns of the table.
name1		The name of the first column
alias1	value of name1	The alias name of the first column. The alias name is used for storing the results of a select statement in a hashtable.
operator1	“and”	The operator between this column and the previous column if they appear in a where clause.
select1		The formula for SELECT action
insert1	the default formula	The formula for INSERT action
update1	the default formula	The formula for UPDATE action
where1	the formula for the UPDATE action	The formula for the value of this column in a where clause. This formula is needed only for an update of a column that is also part of the where clause.
default1		The default formula for all actions except for SELECT
nullValue1		The value for the 1. column in the case that the actual value is SQL NULL.
name2		The name of the second column etc.
alias2	value of name2	The alias name of the second column etc.
nullValue2		The value for the 2. column in the case that the actual value is SQL NULL etc.
maxRowsResultset	0 (endless)	The maximum number of rows for a Resultset after a SELECT command

Syntax of a column name key

```
name<n>=[types.]<name>
```

`<n>`

The number of the column. To be taken into account it must be between 1 and the value of key columns.

`types`

A list of action types of this column:

I	The column will be used for INSERT actions.
S	The column will be used for SELECT actions.

U	The column will be used for UPDATE actions.
C	The column will be used as part of the where clause in SELECT COUNT(*) actions.
D	The column will be used as part of the where clause in DELETE actions.
V	The column will be used as part of the where clause in SELECT actions.
W	The column will be used as part of the where clause in UPDATE actions.
A	The column value will be taken "as is", i.e. it will never embedded in '.

`<name>`

The name of the database column. Table, catalog and schema are omitted, because they are defined with keys name, catalog and schema.

Columns with no action types are considered as "dummy" columns. These columns have not necessarily to exist in the database and will never be accessed. These columns are useful for the substitution process.

`[types.]<name>`

Can contain formulas. See chapter Formulas.

Syntax of a column value key

`select<n>=<column>.<length>=<value>`

or

`where<n>=<value>`

or

`default<n>=<value>`

`select`

is only used for the Putter and ResultWriter programs and for function SELECT. It specifies the format of the output into a file.

`where`

is used if the column is part of a where-clause. If it is omitted then the value of the according default is used.

`default`

is used for all other types of columns.

`<n>`

is the number of the column. To be taken into account it must be between 1 and the value of key columns and the according column name keyword must exist (where <n> is the same).

`<column>`

is the column in the current line of the output file (starting with 1), an empty value indicates the current column.

`<length>`

is the count of characters to write into the current line

`<value>`

contains the value for the column. <value> can contain formulas.

Formulas

A formula is intended to flexible describe the value for the appropriate key.

A formula has the syntax

`${<type>[<name>]<delm>[<function1>][<delm><function2>…]… }`

or in case of non-table properties (only for keys that are marked with (f) in this document)

`${[<name>]<delm>[<function1>][<delm><function2>…]… }`

where the type is implicit assumed to be an H.

These formulas are evaluated before using the appropriate key/value pair. Formulas can be nested. In this case the innermost formulas are evaluated first.

`<type>`

is the type of variable. Uppercase types return String values, lowercase types return byte arrays.

type can be:

- **K**
  The value of a resource key
- **C**
  The value of a column of a table. If the C is not followed by a name then the current column is taken.
- **S**
  The size of a column of a table. If the S is not followed by a name then the current column is taken.
- **Y**
  The SQL type of a column of a table. If the Y is not followed by a name then the current column is taken.
- **F**
  A file
- **L**
  The current input line
- **N**
  An empty string

- **H**

  A value of a hashtable key. There is only one hashtable that can be used for all tables. There are some possibilities to define entries in this table:

  Always defined are:

  VERSION

  The version of the packages service.jar and softpro.jar, delimited by "."

  PROGRAM

  The name of the java-main-class, without preceding package-name

  REPORT

  1 - a report file shall be used

  0 - no report file shall be used

  REPORT.DB

  1 - only those records shall be reported in a report file, that have changed the database

  0 - also those records, that didn’t change the database, shall be reported in a report file

  REPORT.DELIMITER

  The delimiter character to separate the fields in a report record. Default is „,“

  After opening a data file the following entries are defined:

  FILE.NAME

  The name of the data file without path and extension

  FILE.PATH

  The path of the data file without the file name

  FILE.WORKPATH

  The name of the working directory

  FILE.EXTENSION

  The extension of the data file, starting with „.“. If there is no extension, FILE.EXTENSION will be empty

  FILE.SIZE

  The size of the data file

  FILE.DATE

  The date of the last change of the data file in the format

  "yyyy-MM-dd HH:mm:ss.SSS"

  After reading a record from the data file the following entries are defined:

  LINE.NUMBER

  The current line number (starting with 1)

  LINE.CONTENT

  The content of the current line

  FILE.OFFSET

  The current file position

  Whenever a database’s register is queried (defined by specialRegisters and its associated keys), an entry is created with the name of the register as key and the queried value as value.

  Only AccountLoader:

  LAST.TABLE

The name of the last accessed database table.

LAST.RESULT

The number of accessed rows in the current database table.

TABLE

The name of the database table where something has been changed with a successful INSERT, UPDATE or DELETE statement.

REPORT.CHANGED

1 after a successful INSERT, UPDATE or DELETE statement, otherwise 0.

TRAILER

The last record of the data file if the key trailerRecord is true, otherwise TRAILER is empty.

CONTROL.ROW

The number of repetitions of processing a table resource, starting with 1.

Whenever a select is performed on a table, all columns that are read are defined as entries: the column name or the alias of this column, so far defined, will be the key and the value of the column will be the value of the key. If a column NAME is read and the column has an alias ALIAS and the value of this column is VALUE, then the hashtable entry ALIAS with the value VALUE is created.

Only Getter, SignatureReferenceFilter and DataViewer:

AFSDATETIME

After opening an AFS-image-file with function AFS: the DateTime String from the first IFD of the image file.

Only Getter:

PRIMANOTA_NO

This key should contain the current primanota number in case of Primanota-Processing. If this key is not defined, the column PRIMANOTA_NO of the first table will be read instead.

`<name>`

is the name of the variable (file, column, key).

`<delm>`

is the delimiter between variables and functions. There are three possibilities:

delm	Description
?	The following function is not executed if the condition is false. When evaluating a formula, the condition is initially set to undefined, which means an unconditional execution of the following function. Every function can change this condition, but if it doesn't, the condition remains unchanged.
:	The following function is not executed if condition is true.
|	The following function is always executed.

`<function>`

is a function. Functions are executed from left to right. The input value for the function is the output value of its predecessor or the initial value of the variable resp.

Syntax

`<function><parm1>,<parm2>,<parm3>,…`

`<parm>`

- 0…n parameters for the function. If a parameter has the type String then it can be enclosed in double quotes to separate the parameter from the function if this is the first parameter or to use the delimiter comma inside the String. A double quote inside the String is written as 2 double quotes.

`<name>` and `<parm>` can contain the following expressions that are resolved before using:

Expression	Description
$$	The character $.
$*	The String representation of the input value. In case of <name> the input value is empty.
$#	The length of the String representation of the input value. In case of <name> a 0.
$?	The return code of the previous function or 0 for the first function or in case of <name>.
$R	The current reject reason (created e.g. by function REJECT).
$H<key>$	The value of hashtable key <key>. The $ after <key> can be omitted if it is the last character.
$K<key>$	The value of resource file key <key>. The $ after <key> can be omitted if it is the last character.

Predefined functions

This is the list of predefined functions in alphabetical order. It can be extended by using the keys newFunction1, newFunction2, etc. It is also possible, but not recommended, to overwrite an existing function by another one with the same name.

- **ADDONE**

  Return code: 0

  Condition: unchanged

  Description:

  Increases a given timestamp by the smallest possible value.

  Syntax

  `ADDONE”format-pattern”`

  where:

  `format-pattern`

  The format of the timestamp. Default: "yyyy-MM-dd HH:mm:ss.SSSSSS"

- **BIN**
  Return code: 0
  Condition: unchanged
  Description:
  Interprets binary data as a number.
  Syntax
  `BIN[byte-order]`
  where:
  `byte-order`
  0 - Intel byte order
  1 - Motorola byte order
  Default: 0
- **BOOL**
  Return code: unchanged
  Condition: true if the Boolean value is true, otherwise false
  Description:
  Interprets the input value as a Boolean and returns 1 if the result is true, otherwise 0.
  The following strings (case-insensitive) are interpreted as true:
  `true`
  `on`
  `yes`
  `1`
  The following strings (case-insensitive) are interpreted as false:
  `false`
  `off`
  `no`
  `0`
  All other strings provoke a Runtime Exception.
  Syntax
  `BOOL[default]`
  where:
  `default`
  The return value for the case that the input could not be interpreted as a Boolean value. Default: 0
- **BREAK**
  Return code: unchanged
  Condition: unchanged
  Description:
  Finishes the execution of the chain of functions. The following functions are not executed. The value of the formula remains unchanged.
  Syntax
  `BREAK`

- **CALC**

  Return code: 0

  Condition: unchanged

  Description:

  Calculates a number from a numeric expression. The result is an integer value.

  Syntax

  `CALC[expression[,location[,type]]]`

  where:

  `expression`

  A numeric expression starting with one of the operators +, -, *, /, % or SQRT. The first operand is the input value.

  `location`

  The location of the value:

  V - the value as it is. In this case the result becomes the output value

  H - the value is a key of the hashtable containing the actual value. In this case the result is stored as the new value of the hashtable key and the output remains unchanged

  S - the value is the basename of a key in the hashtable. <value>.0 contains the number of values, <value>.1 the first value etc. In this case the results are stored in the same hashtable keys and the output remains unchanged.

  Default: V

  `type`

  The type of the input:

  S - the input is a single numeric value

  L - the input is a comma-separated list of numeric values

  Default: S

- **CASE**

  Return code: unchanged

  Condition: unchanged

  Description:

  Changes the case of the input String, but if the input is a byte array, it does nothing.

  Syntax

  `CASE[modus]`

  where:

  `modus`

  0 - changes nothing

  1 - changes the input to lowercase

  2 - changes the input to uppercase

  3 - invert uppercase and lowercase

  Default: 1

- **CLEAR**
  Return code: The number of deleted keys in the hashtable
  Condition: unchanged
  Description:
  Deletes keys from the hashtable.
  Syntax
  `CLEAR[regex]`
  where:
  `regex`
  A regular expression denoting all keys to be deleted.
  Example
  `CLEAR^SCR-.*`
  Deletes all keys starting with SCR-
- **DATAREAD**
  Return code: 0
  Condition: unchanged
  Description:
  Converts the current date or a given date to an arbitrary format.
  Syntax
  `DATE"[out-time-pattern"[,milliseconds[,in-time-pattern]]]`
  where:
  `out-time-pattern`
  The pattern for the time format (default "yyyy-MM-dd"): To specify the time format use a time-pattern string. In this pattern, all ASCII letters are reserved as pattern letters, which are defined as the following:

```
Symbol Meaning              Presentation    Example
------ -------              ------------    -------
G      era designator       (Text)          AD
y      year                 (Number)        1996
M      month in year        (Text & Number) July & 07
d      day in month         (Number)        10
h      hour in am/pm (1~12) (Number)        12
H      hour in day (0~23)   (Number)        0
m      minute in hour       (Number)        30
s      second in minute     (Number)        55
S      millisecond          (Number)        978
E      day in week          (Text)          Tuesday
D      day in year          (Number)        189
F      day of week in month (Number)        2-2.Wed in May
w      week in year         (Number)        27
W      week in month        (Number)        2
a      am/pm marker         (Text)          PM
k      hour in day (1~24)   (Number)        24
K      hour in am/pm (0~11) (Number)        0
z      time zone            (Text)          Pac. Std. Time
'      escape for text      (Delimiter)
```

```
''      single quote          (Literal)      '
```

The count of pattern letters determine the format.

(Text): 4 or more pattern letters--use full form, <4--use short or abbreviated form if one exists.

(Number): the minimum number of digits. Shorter numbers are

zero-padded to this amount. Year is handled specially; that is, if the count of 'y' is 2, the Year will be truncated to 2 digits.

(Text & Number): 3 or over, use text, otherwise use number.

Any characters in the pattern that are not in the ranges of

['a'..'z'] and ['A'..'Z'] will be treated as quoted text. For instance, characters like ':', '.', ' ', '#' and '@' will appear in the resulting time text even they are not embraced within single quotes.

A pattern containing any invalid pattern letter will result in a thrown exception during formatting or parsing.

`milliseconds`

Before conversion, this value is added to the current time in milliseconds. Default: 0. milliseconds can also be negative.

`in-time-pattern`

If not specified, the time to be converted is the current time.

If specified, the time to be converted is taken from the current value, that is assumed to be a date according to this pattern. The following ASCII letters are supported:

```
Symbol Meaning              Presentation    Example
------ -------              ------------    -------
y      year                 (Number)        1996
M      month in year        (Number)        07 (July)
d      day in month         (Number)        10
h      hour in am/pm (1~12) (Number)        12
H      hour in day (0~23)   (Number)        0
m      minute in hour       (Number)        30
s      second in minute     (Number)        55
S      millisecond          (Number)        978
E      day in week          (Number)        2 (Tuesday)
D      day in year          (Number)        189
w      week in year         (Number)        27
W      week in month        (Number)        2
```

If the in-time-pattern does not specify a time completely, the missing values are taken from the current time.

- **DATEDIFF**
  Return code: 0
  Condition: unchanged
  Description:
  Calculates the difference of 2 dates in a given unit.
  Syntax
  `DATEDIFF"date-string"[,unit[,in-time-pattern]]`
  where:
  `date-string`
  The 2nd operand for building the date difference (the first operand is the input value)
  `unit`
  One of the following units for the date diff result:
  s - second
  m - minute
  h - hour
  d - day
  w - week
  Default: d
  `in-time-pattern`
  The same as in function DATE
  Default: "yyyy-MM-dd"
- **DELETEFILE**
  Return code:
  0 - o.k. or file did not exist
  1 - deleting failed
  2 - name empty
  Condition: unchanged
  Description:
  Deletes a file.
  Syntax
  `DELETEFILE”name”`
  where:
  `name`
  The filename.
- **DIRLIST**
  Return code: unchanged
  Condition: unchanged
  Description:
  Stores the names of files of a directory in the hashtable.
  The input value is supposed to contain the directory of the files.
  Syntax

`DIRLIST”name”,useRegex,stem,subdirs`

where:

`name`

The extension of the files or a regular expression denoting the filename. Default: “.dat”

`useRegex`

1 - use regular expressions for the match.

0 - compare using String.endsWith()

Default: 1

`stem`

The basename in the hashtable. <stem>.0 contains the number of files, <stem>.1 the name of the first file etc. If stem is empty, then nothing will be stored.

`subdirs`

1 - subdirectories are searched

0 - subdirectories are ignored

Default: 0

- **EBCTOASC**

  Return code: unchanged

  Condition: unchanged

  Description:

  Translates from ebcdic to ascii code.

  Syntax

  `EBCTOASC`

  This function is actually only needed if the input file contains texts in different codepages. If this is not the case, the setting codepageDataFile=Cp273 can used instead.

- **ERR**

  Return code: 1

  Condition: unchanged

  Description:

  Finishes the execution of the chain of functions, prints an error message and throws a RuntimeException.

  Syntax

  `ERR[message[,column,value]]`

  where:

  `message`

  The detailed message

  `column`

  The column of a database table

  `value`

  The value for the database table column

- **FIELD**
  Return code: 0
  Condition: unchanged
  Description:
  Takes one field from a chain of fields.
  Syntax
  `FIELD[field-number[,field-separators]]`
  where:
  `field-number`
  The number of the field, starting with 1. Default: 1
  `field-separators`
  A String with all possible field separators. Default: ","
- **FIELDS**
  Return code: 0
  Condition: unchanged
  Description:
  The number of fields
  Syntax
  `FIELDS[field-separators]`
  where:
  `field-separators`
  A String with all possible field separators. Default: ","
- **FILE**
  Return code: 0
  Condition: unchanged
  Description:
  Builds a String using special expressions for the filename of the data file.
  Syntax
  `FILE"[file-format"]`
  where:
  `file-format`
  a formatting String using special expressions for the filename of the data file:
  %v - the drive
  %e - the file extension
  %n - the filename without path and extension
  %p - the path without filename
  %f - the absolute file name
  %% - the % sign
  Default: “%f”
- **FILEINFO**
  Return code: 0

Condition: unchanged for parms Absolute, Canonical, Modified and Length

True for the remaining parms with a positive result

False otherwise

Description:

Returns information about a file. In those cases, where the condition changes, the input remains unchanged.

Syntax

`FILEINFO[“file-property”[,location]]`

where:

`file-property`

A property of a file (only the first character of the property is important):

Exists - Sets condition to true if the file exists, otherwise false.

Read - Sets condition to true if the file is readable, otherwise false.

Write - Sets condition to true if the file is writable, otherwise false.

File - Sets condition to true if the file is a normal file, otherwise false.

Directory - Sets condition to true if the file is a directory, otherwise false.

Canonical - The canonical path of the file.

Absolute - The absolute path of the file

Modified - The time of last modification in milliseconds or 0 if the file does not exist.

Length - The length of the file or 0 if the file does not exist.

`location`

D - the data directory

C - the current directory

- **FMT**

  Return code: 0

  Condition: unchanged

  Description:

  Formats the argument

  Syntax

  `FMT"format-string"[,argument]`

  where:

  `format-string`

  A formatting String according to the C-Library's sprintf format String

  Default: "", with 1 extension:

  S - Substring. This is an extension to the sprintf()

  C - library function. width and precision are the offset and the length of the substring. If the '-' flag was used, the offset is counted from the end of the String, e.g. %-1.2S returns the last 2 characters of the string. The offset is 1-based.

  `argument`

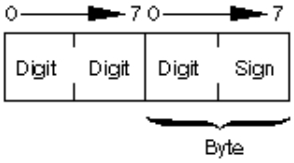
  The argument for the formatting String. If omitted, the input value is used instead.

- **HEX**
  Return code: 0
  Condition: unchanged
  Description:
  Syntax
  `HEX[byte-order]`
  where:
  `byte-order`
  0 - Intel byte order
  1 - Motorola byte order
  Default: 0
- **KEYVALUE**
  Return code: unchanged
  Condition: unchanged
  Description:
  Returns a substring from the input. The substring has to be preceded by the key value and ends before the delimiter value.
  If the key is empty, an empty string is returned.
  Syntax
  `KEYVALUE[key[,delimiter]]`
  where:
  `key`
  The string preceding the desired value. Default: empty
  `delimiter`
  The string following the desired value. If the delimiter was not found in the input string or the delimiter is empty, then the remainder of the input string following the key is returned. Default: empty
  Example
  Input:
  `rc=2; match=17; area=1,2,3,4`
  `KEYVALUE”match=”,”;”`
  Return:
  `“17”`

- **LISTZIP**
  Return code:
  0 - o.k.
  1 - error
  2 - name or basename empty
  Condition: unchanged
  Description:
  Stores the names of the entries of a zip file in the hashtable.
  Syntax
  `LISTZIP”name”[,”stem”]`
  where:
  `name`
  The name of the zip file
  `stem`
  The basename in the hashtable. stem.0 contains the number of entries in the zip file, stem.1 the name of the first entry etc. If stem is empty, then the input value will be the basename.
- **LOAD**
  Return code: unchanged
  Condition: unchanged
  Description:
  Sets the output value.
  Syntax
  `LOAD[source[,name]]`
  where:
  `source`
  One of the sources
  -A a constant
  -C a column of a table
  -L the current input line
  -K a resource file key
  -H a key from the hashtable
  -F a file
  `name`
  The name of the object (redundant for -L) or the value itself in case of -A
  Default: -A

- **LOGIC**
  Return code:
  0 - o.k.
  1 - one of the operators is not numeric
  Condition: unchanged
  Description:
  Performs a logical operation
  Syntax
  `LOGIC[logic-operator[,number]]`
  where:
  `logic-operator`
  One of the operators -AND, -OR, -XOR and -SHIFT
  -SHIFT performs a right shift of the input value if the 2nd operand is positive, otherwise a left shift
  `number`
  The 2nd operand (the first operand is the input value)
  Examples
  `${N|FMT"12"|LOGIC-AND,4} results in 4`
  `${N|FMT"12"|LOGIC-OR,4} results in 12`
  `${N|FMT"12"|LOGIC-XOR,4} results in 8`
- **MESSAGE**
  Return code: unchanged
  Condition: unchanged
  Description:
  Prints a message with trace level in log file. The value of the variable remains unchanged.
  Syntax
  `MESSAGE[[trace-level,]message]`
  where:
  `message`
  The detailed message with trace level.
  Example
  `MESSAGE8,"Imagename=${CIMAGE_NAME}, BITSperPIXEL=$9"`
  `MESSAGE"Image name=${CIMAGE_NAME}, BITSperPIXEL=$9"`
- **PACK**
  Return code: 0
  Condition: unchanged
  Description:
  Packs or unpacks the input value. In case of pack a decimal number is expected, in case of unpack a byte array in packed decimal format.
  Packed-decimal format means that each byte of storage (except for the low order byte) can contain two decimal numbers. The low-order byte contains one digit in the leftmost portion and the sign (positive or

negative) in the rightmost portion. The standard signs are used: hexadecimal F for positive numbers and hexadecimal D for negative numbers. The packed-decimal format looks like this:



The sign portion of the low-order byte indicates whether the numeric value represented in the digit portions is positive or negative.

Syntax

`PACK"direction"`

where:

`direction`

0 - unpack

1 - pack unsigned

3 - pack signed

Default: 3

- **READFILE**

  Return code:

  0 - o.k.

  1 - error

  2 - name empty

  Condition: unchanged

  Description:

  Reads the content of a file line by line and stores the lines in the hashtable.

  Syntax

  `READFILE"name"[,stem]`

  where:

  `name`

  The filename

  `stem`

  The basename in the hashtable. stem.0 contains the number of records that have been read, stem.1 the first record etc. If stem is empty, then the current value of the column will be the basename

- **REJECT**

  Return code: unchanged

  Condition: unchanged

  Description:

  Rejects the current record, i.e. the current record is written to a reject file, so far defined, and no records are written to the database.

  Syntax

  `REJECT[reject-reason,[trace level]]`

  where:

  `reject-reason`

  The comment to the reject. Default: $R

  `trace level`

  The trace level for the log message. Default: ERROR (1)

- **RENAMEFILE**

  Return code:

  0 - o.k.

  1 - error

  2 - from empty

  3 - to empty

  4 - from does not exist

  5 - to does not exist

  Condition: unchanged

  Description:

  Renames a file.

  Syntax

  `RENAMEFILE”from”,”to”[,flag]`

  where:

  `from`

  The current filename

  `to`

  The new filename

  `flag`

  1 - if error, write a warning into the log

  0 - don’t write warnings

  Default is 0.

- **REPLACE**

  Return code: unchanged

  Condition: unchanged

  Description:

  Replaces each substring of the input string that matches the given regular expression with the given replacement.

  Syntax

  `REPLACE[regex,[replacement]]`

  where:

  `regex`

  The regular expression to which the input string is to be matched. Default: empty

  `replacement`

  The string to be substituted for each match. Default: empty
- **RESOURCEKEY**

  Return code: unchanged

  Condition: true if a key was found, otherwise false

  Description:

  Returns the first key of the current resource that matches the regular expression regex or an empty String if no key matches.

  Syntax

  `RESOURCEKEY[regex]`

  where:

  `regex`

  The regular expression to which a key of the current resource is to be matched. Default: empty

- **SAVE**

  Return code: unchanged

  Condition: unchanged

  Description:

  Writes the value value (or the input value if value is not defined) in the hashtable. The value of the variable remains unchanged.

  Syntax

  `SAVE"name"[,value[,location]]`

  where:

  `name`

  The name for this value.

  `value`

  The value for name.

  `location`

  The location of the value:

  V - the value as it is

  K - the value is a key of the resource file containing the actual value

  H - the value is a key of the hashtable containing the actual value

  Default: V
- **SEARCHWORD**

  Return code: unchanged

  Condition: unchanged

  Description:

  Returns the first three characters of a word in the input string. A word in this sense is a number of non-delimiting characters delimited by delimiting characters. These are all characters below 0x80 not being letters or digits.

  If the desired word does not exist, an empty string is returned.

  If the found word has less than 4 characters, the whole word is returned.

  Syntax

  `SEARCHWORD[index]`

  where:

  `index`

  The number of the word, starting with 1. Default: 1
- **SIZE**

  Return code: 0

  Condition: unchanged

  Description:

  The size of the input in bytes

  Syntax

  `SIZE`

- **STEMINDEX**

  Return code: unchanged

  Condition: unchanged

  Description:

  Returns the index beginning with 1 of the found value in the given stem of values in the hashtable or 0 if not found.

  Syntax

  `STEMINDEX"stem"`

  where:

  `stem`

  The basename in the hashtable. <stem>.0 contains the number of values, <stem>.1 the first value etc. If stem is empty, then "0" is returned. Default: empty
- **SUB**

  Return code:

  0 on success

  -1 otherwise

  Condition: unchanged

  Description:

  Substring

  Syntax

  `SUB[from-index!to-index[,modus]]`

  or

  `SUB[from-index.length[,modus]]`

  where:

  `from-index`

  The beginning index. Default: 1

  `to-index`

  The ending index. Default: the length of the string (- 1)

  `length`

  The length of the substring. Default: the remaining length

  `modus`

  The substring modus:

  0 - remove leading and trailing whitespaces

  1 - keep leading whitespaces

  2 - keep trailing whitespaces

  4 - be tolerant with from-index, to-index and length if they are completely or partially outside the substrings boundaries.

  Default: 4

  Combinations are achieved by adding their values. A removing of whitespaces is only performed when the result is a String from-index, to-index are 1-based. A negative value counts backwards from the end of the value, eg. -1 is the last index.
- **TEST**

Return code: unchanged

Condition: true if the test condition matches, otherwise false

Description:

An expression is evaluated. The value of the variable is not changed. If the expression evaluates to true then the following functions preceded by : are ignored. If the expression evaluates to false then the following functions preceded by ? are ignored.

Syntax

```
TEST"value1"operator"value2"
```

where:

`value1,value2`

The values to be compared, either String or number.

`operator`

One of the following operators:

< less than

== equal

<= less than or equal

> greater than

!= not equal

>= greater than or equal

- **TOBINARY**

  Return code: unchanged

  Condition: unchanged

  Description:

  Returns a byte array containing the input as a binary number

  Syntax

  ```
  TOBINARY[digits[,radix]]
  ```

  where:

  `digits`

  The number of bytes of the output. Default: 1

  `radix`

  The radix of the number. Default: 10 (a decimal number)

- **TOBYTE**
  Return code: unchanged
  Condition: unchanged
  Description:
  Returns a byte array containing the input String as a array of bytes
  Syntax
  `TOBYTE[codepage[,trailing-zero-bytes]]`
  where:
  `codepage`
  The codepage to be used for the translation. Default: UTF-8
  `trailing-zero-bytes`
  The number of trailing zero-bytes (to be able to be C-String compliant). Default: 1
- **TRANSLATE**
  Return code: unchanged
  Condition: unchanged
  Description:
  Changes all bytes, whose values are found in a list, to other values or removes these bytes.
  Syntax
  `TRANSLATE[from[,to]]`
  `from`
  A hexadecimal list of values.
  `to`
  A 2nd hexadecimal list of values.
  Every value, that is found in the from-list, will be replaced by the according value in the to-list. Is there no such according value in the to-list, because this list is shorter than the from-list, this byte will be removed.
  Example
  `current value is ABCDEB (hex 414243444542)`
  TRANSLATE4243,41 changes the value to AADEA

- **WARN**

  Return code: unchanged

  Condition: unchanged

  Description:

  Prints a warning message. The value of the variable remains unchanged. This function is the same as the function MESSAGE with trace-level=2

  Syntax

  `WARN[message]`

  where:

  `message`

  The detailed message

  Example

  `WARN"Image name=${CIMAGE_NAME}, BITSperPIXEL=$HBPP"`

- **WRITEFILE**

  Return code:

  0 - o.k.

  1 - error

  2 - name empty

  Condition: unchanged

  Description:

  Writes text line by line in a file. If the file already exists, the test will be appended.

  Syntax

  `WRITEFILE”name”,”value”[,mode]`

  where:

  `name`

  The filename

  `value`

  The text to be written. If value is empty, the current value of the column will be written.

  `mode`

  H - value is the name of a key from the hashtable. Its value will be written

  S - value is the basename of a key from the hashtable. <value>.0 lines will be written, the first line is <value>.1 etc.

  Otherwise the value itself will be written.

- **UNZIP**

  Return code:

  0 - o.k.

  1 - error

  2 - name empty

  3 - entry not found

  Condition: unchanged

  Description:

  Reads the content of a zip file entry. If the return code is not 0, an error message will be returned instead of the file content.

  Syntax

  `UNZIP"name"[,"entry"]`

  where:

  `name`

  The name of the zip file.

  `entry`

  The name of the entry that is to read. If entry is empty, then the current value of the column will be the name of the entry.

Additional functions

It is possible to add functions to the set of existing functions described above. This is achieved by defining the key newFunctions and its associated keys in the main properties file.

Currently the following functions are known: (parameters are delimited by ":" or by the value defined with key newFunction<n>Delimiter, where <n> is the number of the new function).

- **AFS**

  Class name: de.softpro.signplus.service.AFSImage
- **AFSX**

  Class name: de.softpro.signplus.service.AFSx937
- **CIFF**

  Class name: de.softpro.signplus.service.CIFF

  Parameters:

  1. Number of leading bytes per record (4 or 0).
- **CHKG**

  Class name: de.softpro.signplus.service.AFSImage

  Components needed at runtime: getter.dll, LeadTools

- **CHECKHITRATE**

  Class name: de.softpro.signplus.service.CheckHitrate

  Parameters:

  1. Name of the resource containing the name of the reference evaluator class.

  2. Name of the key containing the name of the reference evaluator class. This class must be derived from de.softpro.signplus.service.CheckHitrate. If resource, key or reference evaluator class does not exist, de.softpro.signplus.service.CheckHitrate is used as reference evaluator class.

  3. Base name of the stem in the hashtable containing the image numbers of the references to be evaluated.

  4. Base name of the stem in the hashtable containing the COUNTERUSED values of the references.

  5. Base name of the stem in the hashtable containing the VALIDFROM dates of the references.

  6. Base name of the stem in the hashtable containing the VALIDTO dates of the references.

  All base names are expected to have the same count of entries, i.e. <basename>.0 is identical for all base names.
- **CHECKRANGE**

  Class name: de.softpro.signplus.service.CheckRange

  Parameters:

  1. Name of the resource containing the list of ranges.

  2. Base name of the key containing the list of ranges. <basename>.0 contains the count of ranges, <basename>.1 contains the first (amount) range, comma separated (in cent) etc.
- **CHECKRANGECORPORATE**

  Class name: de.softpro.signplus.service.CheckRange

  Parameters:

  1. Name of the resource containing the list of ranges.

  2. Base name of the key containing the list of ranges. <basename>.0 contains the count of ranges, <basename>.1 contains the first (amount) range, comma separated (in cent) etc.
- **CHECKRANGEOTHER**

  Class name: de.softpro.signplus.service.CheckRange

  Parameters:

  1. Name of the resource containing the list of ranges.

  2. Base name of the key containing the list of ranges. <basename>.0 contains the count of ranges, <basename>.1 contains the first (amount) range, comma separated (in cent) etc.
- **CHECKRANGEPRIVATE**

  Class name: de.softpro.signplus.service.CheckRange

  Parameters:

  1. Name of the resource containing the list of ranges.

  2. Base name of the key containing the list of ranges. <basename>.0 contains the count of ranges, <basename>.1 contains the first (amount) range, comma separated (in cent) etc.
- **CLEAN**

  Class name:

  Components needed at runtime: getter.dll, spjdec.dll, SicProper

- **CLIP**
  Class name: de.softpro.signplus.service.ClipFunction
  Components needed at runtime: getter.dll, LeadTools
- **CMPEXT**
  Class name: de.softpro.signplus.service.CompareExtensions
- **COUNTPIXELS**
  Class name: de.softpro.signplus.service.CountPixels
  Components needed at runtime: getter.dll, LeadTools
- **CSL**
  Class name: de.softpro.signplus.service.CheckSignatureLicence
  Parameters:
  1. Action, “early” or “late”, determines the time to retrieve the license count. Default: “late”
  Components needed at runtime: jpwenc.dll, splm2
- **GRAYCLEANZIP**
  Class name: de.softpro.signplus.service.GrayCleanZipFunction
  Components needed at runtime: getter.dll, SPUtils.zip, SicProper
- **IFD**
  Class name: de.softpro.signplus.service.IFD
  Components needed at runtime: tiff.jar
- **LINESEARCH**
  Class name: de.softpro.signplus.service.LineSearch
  Components needed at runtime: getter.dll, LeadTools
- **LTIF**
  Class name: de.softpro.signplus.service.LoadTiffFileFunction
  Parameters:
  1. The number of simultaneously open files. Default: 50
  Components needed at runtime: tiff.jar
- **MONOCLEAN**
  Class name: de.softpro.signplus.service.CleanSival
  Parameters:
  1. Name of the resource file containing the tracelevels for sival messages. Default: no resource file, all sival messages are logged with trace level DEBUG
  Components needed at runtime: jsival.dll, Sival

- **PAD**

  Class name: de.softpro.signplus.service.GIAPAD

  Parameters:

  padResource - name of the resource file containing the PAD keys

  padKey - base name of the PAD keys. <padKey>.0 contains the count of keys, <padKey>.1 is the 1st key <padKey>.2 is the 2nd key etc.

  blacklistResource - name of the resource containing the blacklist keys. The count of keys is stored in the key “999.Name.0”, the first key is in “999.Name.1”, the second key in “999.Name.2” etc.

  Components needed at runtime: PAD_APIA.dll, jApia.dll
- **SBSTAT**

  Class name: de.softpro.signplus.service.SBStatistics

  Components needed at runtime:
- **SELECT**

  Class name: de.softpro.signplus.service.ProcessSPTable

  Parameters: Name of the table properties file to perform the SELECT.

  1. Components needed at runtime:
- **SIMPLICITY**

  Class name: de.softpro.signplus.service.SivalSimplicity

  Parameters:

  1. Name of the resource file containing the tracelevels for sival messages. Default: no resource file, all sival messages are logged with trace level DEBUG.

  Components needed at runtime: jsival.dll, Sival
- **SIVALCOMPARE**

  Class name: de.softpro.signplus.service. SivalCompare

  Parameters:

  1. Name of the resource file containing the tracelevels for sival messages. Default: no resource file, all sival messages are logged with trace level DEBUG

  Components needed at runtime: jsival.dll, Sival

- **SIZETYPE**
  Class name: de.softpro.signplus.service.
  Parameters:
  1. Name of the resource file containing the following keys:
  `resolution`
  x- and y-resolution, comma-separated. Default: 300,300
  `default`
  The size type if none of the ranges matches. Default: 0
  `range`
  A stem containing ranges. Syntax of a range:
  `X|Y from [ - until ] [ X|Y from [ - until ] ] = sizetype`
  where:
  `X|Y`
  is either X or Y coordinate
  `from`
  is the minimum value
  `until`
  is the maximum value
  `sizetype`
  is the sizetype for this range. If one of the coordinates has been omitted, this coordinate will not be checked.
- **UNZIPP**
  Class name: de.softpro.signplus.client.scclient.SP_UnzipFunction
  Components needed at runtime: spjdec.dll
- **VARIANTCOMPARE**
  Class name: de.softpro.signplus.service.VariantCompare
  Parameters:
  1. Name of the resource file containing the tracelevels for sival messages.
  Default: no resource file, all sival messages are logged with trace level DEBUG
  Components needed at runtime: jsival.dll, Sival
- **(not defined)**
  Class name: de.softpro.signplus.service.FunctionDummy
  Parameters: (as many as needed)

The last (not defined) function serves as dummy function to change the functionality of a function without changing anything in the configuration. E.g. if you don’t want to use an external software to check for a PAD you just have to replace the setting:

```
newFunction10 =
de.softpro.signplus.service.PAD:PAD:service:Getter.PADkey:PayeeList
```

with

```
newFunction10 =
de.softpro.signplus.service.FunctionDummy:PAD:service:Getter.PADkey:PayeeList
```

And here is their description:

- **AFS**

  Return code: unchanged

  Condition: unchanged

  Description:

  Returns a single TIFF image from an AFS image file. The input value is considered to be the sequence number of the image.

  Syntax

  `AFS"side","PI"`

  where:

  `side`

  F - the front image

  B - the back image

  `PI (Photometric Interpretation)`

  0 - Photometric Interpretation is set to 0

  1 - Photometric Interpretation is set to 1

  Empty Photometric Interpretation remains unchanged (default)

  Whenever an image file is opened, its date/time string is stored in the hashtable key AFSDATETIME.

- **AFSX**
  Return code: unchanged
  Condition: true if a record was found, otherwise false
  Description:
  Returns the number of found records (0 or 1)in the input data file which is assumed to be an AFX x9.37 file. The input is ignored.
  Syntax
  `AFSX"basename"`
  where:
  `basename`
  is the base name for following keys created in the hashtable. Default: AFSx937
  AFSx937.R20.CLEARDATE
  Bundle Business Date
  AFSx937.R20.CLEARDATE
  Bundle Business Date
  AFSx937.R25.SERIALNO
  Auxiliary On-Us
  AFSx937.R25.PROCODE
  External Processing Code
  AFSx937.R25.ROUTING
  Payor Bank Routing Number
  AFSx937.R25.CHECKDIGIT
  Payor Bank Routing Check Digit
  AFSx937.R25.ACCOUNT
  On-Us
  AFSx937.R25.AMOUNT
  Item Amount
  AFSx937.R25.SEQUENCE
  ECE Institution Item Sequence Number
  AFSx937.R26.RECORDS
  the number of records with type 26
  AFSx937.R26.ROUTING
  Bank of First Deposit (BOFD) Routing Number
  AFSx937.R26.DATE
  BOFD Business (Endorsement) Date
  AFSx937.R26.SEQUENCE
  BOFD Item Sequence Number
  AFSx937.R26.ACCOUNT
  Deposit Account Number at BOFD
  AFSx937.R26.BRANCH
  BOFD Deposit Branch

AFSx937.R26.PAYEE
Payee Name
AFSx937.R26.TRUNC
Truncation Indicator
AFSx937.R26.BOFDCON
BOFD Conversion Indicator
AFSx937.R26.BOFDCOR
BOFD Correction Indicator
AFSx937.R26.ROUTING
Bank of First Deposit (BOFD) Routing Number
AFSx937.R26.DATE.n
BOFD Business (Endorsement) Date
AFSx937.R26.SEQUENCE.n
BOFD Item Sequence Number
AFSx937.R26.ACCOUNT.n
Deposit Account Number at BOFD
AFSx937.R26.BRANCH.n
BOFD Deposit Branch
AFSx937.R26.PAYEE.n
Payee Name
AFSx937.R26.TRUNC.n
Truncation Indicator
AFSx937.R26.BOFDCON.n
BOFD Conversion Indicator
AFSx937.R26.BOFDCOR.n
BOFD Correction Indicator
AFSx937.R28.RECORDS
the number of records with type 28
AFSx937.R28.ROUTING
Endorsing Bank Routing Number
AFSx937.R28.DATE
Endorsing Bank Endorsement Date
AFSx937.R28.SEQUENCE
Endorsing Bank Item Sequence Number
AFSx937.R28.ROUTING.n
Endorsing Bank Routing Number
AFSx937.R28.DATE.n
Endorsing Bank Endorsement Date
AFSx937.R28.SEQUENCE.n
Endorsing Bank Item Sequence Number

AFSx937.R50.SIDE
the current side of the cheque, 0=front, 1=back
AFSx937.R50.ROUTING.n
Image Creator Routing Number
AFSx937.R50.DATE.n
Image Creator Date
AFSx937.R50.DATASIZE.n
Image View Data Size
AFSx937.R52.ROUTING.n
ECE Institution Routing Number
AFSx937.R52.DATE.n
Bundle Business Date
AFSx937.R52.CYCLE.n
Cycle Number
AFSx937.R52.SEQUENCE.n
ECE Institution Item Sequence Number
AFSx937.R52.IMAGEOFFSET.n
Offset of the Image inside the whole File
AFSx937.R52.IMAGE.n
Image Data Auxiliary On-Us
AFSx937.R25.PROCODE
External Processing Code
AFSx937.R25.ROUTING
Payor Bank Routing Number
AFSx937.R25.CHECKDIGIT
Payor Bank Routing Check Digit
AFSx937.R25.ACCOUNT
On-Us
AFSx937.R25.AMOUNT
Item Amount
AFSx937.R25.SEQUENCE
ECE Institution Item Sequence Number
AFSx937.R26.RECORDS
the number of records with type 26
AFSx937.R26.ROUTING
Bank of First Deposit (BOFD) Routing Number
AFSx937.R26.DATE
BOFD Business (Endorsement) Date
AFSx937.R26.SEQUENCE
BOFD Item Sequence Number

AFSx937.R26.ACCOUNT
Deposit Account Number at BOFD
AFSx937.R26.BRANCH
BOFD Deposit Branch
AFSx937.R26.PAYEE
Payee Name
AFSx937.R26.TRUNC
Truncation Indicator
AFSx937.R26.BOFDCON
BOFD Conversion Indicator
AFSx937.R26.BOFDCOR
BOFD Correction Indicator
AFSx937.R26.ROUTING
Bank of First Deposit (BOFD) Routing Number
AFSx937.R26.DATE.n
BOFD Business (Endorsement) Date
AFSx937.R26.SEQUENCE.n
BOFD Item Sequence Number
AFSx937.R26.ACCOUNT.n
Deposit Account Number at BOFD
AFSx937.R26.BRANCH.n
BOFD Deposit Branch
AFSx937.R26.PAYEE.n
Payee Name
AFSx937.R26.TRUNC.n
Truncation Indicator
AFSx937.R26.BOFDCON.n
BOFD Conversion Indicator
AFSx937.R26.BOFDCOR.n
BOFD Correction Indicator
AFSx937.R28.RECORDS
the number of records with type 28
AFSx937.R28.ROUTING
Endorsing Bank Routing Number
AFSx937.R28.DATE
Endorsing Bank Endorsement Date
AFSx937.R28.SEQUENCE
Endorsing Bank Item Sequence Number
AFSx937.R28.ROUTING.n
Endorsing Bank Routing Number

AFSx937.R28.DATE.n
Endorsing Bank Endorsement Date
AFSx937.R28.SEQUENCE.n
Endorsing Bank Item Sequence Number
AFSx937.R50.SIDE
the current side of the cheque, 0=front, 1=back
AFSx937.R50.ROUTING.n
Image Creator Routing Number
AFSx937.R50.DATE.n
Image Creator Date
AFSx937.R50.DATASIZE.n
Image View Data Size
AFSx937.R52.ROUTING.n
ECE Institution Routing Number
AFSx937.R52.DATE.n
Bundle Business Date
AFSx937.R52.CYCLE.n
Cycle Number
AFSx937.R52.SEQUENCE.n
ECE Institution Item Sequence Number
AFSx937.R52.IMAGEOFFSET.n
Offset of the Image inside the whole File
AFSx937.R52.IMAGE.n
Image Data

- **CIFF**

  Return code: unchanged
  Condition: true if a record was found, otherwise false
  Description:
  Returns the number of found records (0 or 1)in the input data file which is assumed to be an CIFF file. The input is ignored.
  Syntax
  `AFSX"basename"`
  where:
  `basename`
  is the base name for following keys created in the hashtable. Default: Ciff
  Ciff.R1.VERSION
  Record format version
  Ciff.R1.FORMAT
  Format of text and binary data, 1=EBCDIC
  Ciff.R1.CODEPAGE
  Codepage used to generate this CIFF

Ciff.R1.ORIGIN
Origin identification
Ciff.R1.DATE
CIFF creation date (yyyyMMdd)
Ciff.R1.TIME
CIFF creation time (HHmmssSS)
Ciff.R1.TRACK
Volume tracking Identification
Ciff.R1.CATEGORY
Category or type of images
Ciff.R10.VERSION
Record format version
Ciff.R10.KEYCONFLICT
Key Conflict option
Ciff.R10.DATE
Cycle Date option (yyyyMMdd)
Ciff.R10.SORTER
Sorter option
Ciff.R10.PREFIX
Sequence Number Prefix option
Ciff.R10.REPOSITORY
Target Repository option
Ciff.R10.WRAPPER
Wrapper Conversion option
Ciff.R10.CIMSRETCODES
number of CIMS 604 return codes
Ciff.R15.VERSION
Record format version
Ciff.R30.VERSION
Record format version
Ciff.R30.<name>
0 to n index records, known names are: XAMOUNT, XPC, XACCT, XRT, XSERNO
Ciff.R40.VERSION
Record format version
Ciff.R40.CODED_DATA
Coded data
Ciff.R40.FRONTIMAGE
Front Image b/w
Ciff.R40.FRONTIMAGE-G
Front Image grayscale

Ciff.R40.BACKIMAGE
Back Image b/w
Ciff.R40.BACKIMAGE-G
Back Image grayscale
Ciff.R40.SEGMENT5
segment 5
Ciff.R40.SEGMENT6
segment 6
Ciff.R40.SEGMENT7
segment 7
Ciff.R40.SEGMENT8
segment 8
Ciff.R40.SEGMENT9
segment 9
Ciff.R40.SEGMENT10
segment 10
Ciff.R40.SEGMENT11
segment 11
Ciff.R40.SEGMENT12
segment 12
Ciff.R40.SEGMENT13
segment 13
Ciff.R40.SEGMENT14
segment 14
Ciff.R40.SEGMENT15
segment 15
Ciff.RFF.IMGCNT
number of items

- **CHKG**

  Return code: on success 1 on error

  Condition: true if the byte array contains an image, otherwise false

  Description:

  Checks the input byte array whether it contains an image or not

  Syntax

  `CHKG`

  CHKG creates the following hashtable entries, where <column> is the name of the column (or its alias) where CHKG is specified:

  `<column>.RC`

  0 o.k. otherwise the input contains no image

  `<column>.WIDTH`

  Width

<column>.HEIGHT

Height

<column>.BPP

Bits Per Pixel

<column>.XRES

X-Resolution (integer)

<column>.YRES

Y-Resolution (integer)

<column>.PAGES

Number of Pages

<column>.FORMAT

the format of the image:

```
DAV 0   App Informatik SignCheck Format
PCX 1   Zsoft PCX
GIF 2   CompuServe GIF
TIF 3   Tagged Image File Format
TGA 4   Targa
CMP 5   LEAD CMP
BMP 6   Windows BMP
JFIF 10   Jpeg File Interchange Format
JTIF 11   Jpeg Tag Image File Format
OS2 14    OS/2 BMP
WMF 15    Windows Meta File
EPS 16    Encapsulated Post Script
TIFLZW 17    TIF Format with LZW compression
LEAD 20      LEAD Proprietary
LEAD1JFIF 21    JPEG 4:1:1
LEAD1JTIF 22    JPEG 4:1:1
LEAD2JFIF 23    JPEG 4:2:2
LEAD2JTIF 24    JPEG 4:2:2
CCITT 25      TIFF CCITT
LEAD1BIT 26   LEAD 1 bit, lossless compression
CCITT_GROUP3_1DIM 27   CCITT Group3 one dimension
CCITT_GROUP3_2DIM 28   CCITT Group3 two dimensions
CCITT_GROUP4 29     CCITT Group4 two dimensions
LEAD_NOLOSS 30      LEAD Proprietary Lossless
LEAD1BITA 31        old LEAD 1 bit, lossless compression
CALS 50   CALS
MAC 51    MAC
IMG 52    IMG
MSP 53    MSP
WPG 54    WPG
RAS 55    RAS
PCT 56    PCT
PCD 57    PCD
DXF 58    DXF
AVI 59    AVI
WAV 60    WAV
FLI 61    FLI
CGM 62    CGM
EPSTIFF 63   EPS with TIFF Preview
EPSWMF 64    EPS with Metafile Preview
CMPNOLOSS 65    CMPNOLOSS
FAX_G3_1D 66    FAX_G3_1D
FAX_G3_2D 67    FAX_G3_2D
FAX_G4 68       FAX_G4
```

```
WFX_G3_1D 69    WFX_G3_1D
WFX_G4 70       WFX_G4
ICA_G3_1D 71    ICA_G3_1D
ICA_G3_2D 72    ICA_G3_2D
ICA_G4 73       ICA_G4
OS2_2 74        OS2_2
PNG 75    PNG
PSD 76    PSD
RAWICA_G3_1D 77   RAWICA_G3_1D
RAWICA_G3_2D 78   RAWICA_G3_2D
RAWICA_G4 79      RAWICA_G4
FPX 80     FlashPix, no compression
FPX_SINGLE_COLOR 81     FlashPix, compression 'single color' method
FPX_JPEG 82      FlashPix, compression JPEG
FPX_JPEG_QFACTOR 83 FlashPix, compression JPEG, specify qFactor
BMP_RLE 84      compressed Windows BMP
TIF_CMYK 85     TIFF no compression, CMYK data
TIFLZW_CMYK 86         TIFF LZW compression, CMYK data
TIF_PACKBITS 87        TIFF PackBits compression, RGB data
TIF_PACKBITS_CMYK 88   TIFF PackBits compression, CMYK data
DICOM_GRAY 89    DICOM_GRAY
DICOM_COLOR 90   DICOM_COLOR
WIN_ICO 91       WIN_ICO
WIN_CUR 92       WIN_CUR
TIF_YCC 93       TIFF YcbCr color space, no compression
TIFLZW_YCC 94    TIFF YcbCr color space, LZW compression
TIF_PACKBITS_YCC 95    TIFF YcbCr color space, PackBits compression
EXIF 96        uncompressed RGB Exif file
EXIF_YCC 97    uncompressed YcbCr Exif file
EXIF_JPEG 98   JPEG compressed Exif file
AWD 99         Microsoft Fax format
FASTEST 100    for ISIS only! use the data as is, from the ISIS Scanner
```

- **CHECKHITRATE**

  Return code: index with the lowest hitrate that is worth to be deleted or 0 when there is no such reference

  Condition: unchanged

  Description:

  Checks the hitrate (column COUNTER_USED) of a stem of references defined in the constructor of this function. The index with the lowest hitrate that is worth to be deleted is returned, starting with 1, or 0 when there is no such reference. The input is ignored.

  Syntax

  `CHECKHITRATE[min-age,[min-count]]`

  where:

  `min-age`

  is the minimum age of the references in days,

  `min-count`

  is the minimum count of valid references.

- **CHECKRANGE**

  Return code: 0

  Condition: unchanged

  Description:

  Returns the index of the first range where the input value is inside, starting with 1, or 0 when value is outside of all ranges.

  Syntax

  `CHECKRANGE`
- **CHECKRANGECORPORATE**

  Return code: 0

  Condition: unchanged

  Description:

  Returns the index of the first range where the input value is inside, starting with 1, or 0 when value is outside of all ranges.

  Syntax

  `CHECKRANGECORPORATE`
- **CHECKRANGEOTHER**

  Return code: 0

  Condition: unchanged

  Description:

  Returns the index of the first range where the input value is inside, starting with 1, or 0 when value is outside of all ranges.

  Syntax

  `CHECKRANGEOTHER`
- **CHECKRANGEPRIVATE**

  Return code: 0

  Condition: unchanged

  Description:

  Returns the index of the first range where the input value is inside, starting with 1, or 0 when value is outside of all ranges.

  Syntax

  `CHECKRANGEPRIVATE`

- **CLEAN**
  Return code: unchanged
  Condition: unchanged
  Description:
  Cleans a grayscale Tiff Image and converts it to a dedicated format.
  Syntax
  `CLEAN[clean-level,[depth]]`
  where:
  `clean-level`
  is the level of cleaning (hexadecimal):
  00000ii ii=index of proper fine tuning parameter set
  0000100 NOISE
  0000200 BACKGROUND
  0000400 TEXTURE
  0000800 LINE
  0001000 STAMP
  0002000 RECONSTRUCT
  0004000 CLEANFRAME
  0010000 NONE
  0020000 HOMOGEN
  4000000 PSEUDOCOLOR
  8000000 AUTOMATIC
  If no bit is set, no cleaning takes place.
  The levels can be combined by adding their values (hexadecimal). Default: 8022000
  `depth`
  is the number of bits per pixels:
  1 = 1 bit per pixel, format CCITT_GROUP4
  0 = leave the format of the image unchanged
  Default: 1
  CLEAN creates the following hashtable entries, where <column> is the name of the column (or its alias) where CLEAN is specified:
  `<column>.RC`
  0 o.k. otherwise the input contains no image
  `<column>.WIDTH`
  Width
  `<column>.HEIGHT`
  Height
  `<column>.BPP`
  Bits Per Pixel
  `<column>.XRES`
  X-Resolution (integer)

<column>.YRES

Y-Resolution (integer)

<column>.PAGES

Number of Pages

<column>.FORMAT

the format of the image:

```
DAV 0      App Informatik SignCheck Format
PCX 1      Zsoft PCX
GIF 2      CompuServe GIF
TIF 3      Tagged Image File Format
TGA 4      Targa
CMP 5      LEAD CMP
BMP 6      Windows BMP
JFIF 10    Jpeg File Interchange Format
JTIF 11    Jpeg Tag Image File Format
OS2 14     OS/2 BMP
WMF 15     Windows Meta File
EPS 16     Encapsulated Post Script
TIFLZW 17   TIF Format with LZW compression
LEAD 20     LEAD Proprietary
LEAD1JFIF 21   JPEG 4:1:1
LEAD1JTIF 22   JPEG 4:1:1
LEAD2JFIF 23   JPEG 4:2:2
LEAD2JTIF 24   JPEG 4:2:2
CCITT 25       TIFF CCITT
LEAD1BIT 26    LEAD 1 bit, lossless compression
CCITT_GROUP3_1DIM 27    CCITT Group3 one dimension
CCITT_GROUP3_2DIM 28    CCITT Group3 two dimensions
CCITT_GROUP4 29    CCITT Group4 two dimensions
LEAD_NOLOSS 30     LEAD Proprietary Lossless
LEAD1BITA 31       old LEAD 1 bit, lossless compression
CALS 50   CALS
MAC 51    MAC
IMG 52    IMG
MSP 53    MSP
WPG 54    WPG
RAS 55    RAS
PCT 56    PCT
PCD 57    PCD
DXF 58    DXF
AVI 59    AVI
WAV 60    WAV
FLI 61    FLI
CGM 62    CGM
EPSTIFF 63   EPS with TIFF Preview
EPSWMF 64    EPS with Metafile Preview
CMPNOLOSS 65    CMPNOLOSS
FAX_G3_1D 66    FAX_G3_1D
FAX_G3_2D 67    FAX_G3_2D
FAX_G4 68       FAX_G4
WFX_G3_1D 69    WFX_G3_1D
WFX_G4 70       WFX_G4
ICA_G3_1D 71    ICA_G3_1D
ICA_G3_2D 72    ICA_G3_2D
ICA_G4 73       ICA_G4
OS2_2 74    OS2_2
PNG 75      PNG
PSD 76      PSD
RAWICA_G3_1D 77   RAWICA_G3_1D
RAWICA_G3_2D 78   RAWICA_G3_2D
```

```
RAWICA_G4 79      RAWICA_G4
FPX 80               FlashPix, no compression
FPX_SINGLE_COLOR 81   FlashPix, compression 'single color' method
FPX_JPEG 82           FlashPix, compression JPEG
FPX_JPEG_QFACTOR 83   FlashPix, compression JPEG, specify qFactor
BMP_RLE 84       compressed Windows BMP
TIF_CMYK 85      TIFF no compression, CMYK data
TIFLZW_CMYK 86   TIFF LZW compression, CMYK data
TIF_PACKBITS 87  TIFF PackBits compression, RGB data
TIF_PACKBITS_CMYK 88     TIFF PackBits compression, CMYK data
DICOM_GRAY 89   DICOM_GRAY
DICOM_COLOR 90  DICOM_COLOR
WIN_ICO 91   WIN_ICO
WIN_CUR 92   WIN_CUR
TIF_YCC 93   TIFF YcbCr color space, no compression
TIFLZW_YCC 94         TIFF YcbCr color space, LZW compression
TIF_PACKBITS_YCC 95   TIFF YcbCr color space, PackBits compression
EXIF 96   uncompressed RGB Exif file
EXIF_YCC 97    uncompressed YcbCr Exif file
EXIF_JPEG 98   JPEG compressed Exif file
AWD 99   Microsoft Fax format
FASTEST 100   for ISIS only! use the data as is, from the ISIS Scanner
```

- **CLIP**

  Return code: unchanged

  Condition: unchanged

  Description:

  Cut a rectangle from an Image readable by LeadTools.

  Syntax

  ```
  CLIP[left-formula[,top-formula[,right-formula[,bottom-
  formula[,format[,depth[,resolution]]]]]]]
  ```

  where:

  `xxxx-formula`

  is the formula for finding the xxxx edge (xxxx=left, top,right or bottom resp.):

  `xxxx[-length]`

  or

  `xxxx[+seek[+|-indent]]`

  `left`

  is the pixel coordinate of the left edge, counted from the left edge of the image. A negative value counts from the right edge of the image. Default: 1

  `top`

  is the pixel coordinate of the top edge, counted from the top edge of the image. A negative value counts from the bottom edge of the image. Default: 1

  `right`

  is the pixel coordinate of the right edge, counted from the right edge of the image. A negative value counts from the left edge of the image. Default: -1

  `bottom`

  is the pixel coordinate of the bottom edge, counted from the bottom edge of the image. A negative value counts from the top edge of the image. Default: -1

  `length`

  is the width resp. height of the rectangle if a line was found.

  `seek`

  seek for a line within the next seek pixels.

  `indent`

  Add this value to the position of a found line.

  `format`

  The format of the clipped image as String and numeric:

  ```
  DAV 0     App Informatik SignCheck Format
  PCX 1     Zsoft PCX
  GIF 2     CompuServe GIF
  TIF 3     Tagged Image File Format
  TGA 4     Targa
  CMP 5     LEAD CMP
  BMP 6     Windows BMP
  JFIF 10   Jpeg File Interchange Format
  JTIF 11   Jpeg Tag Image File Format
  OS2 14    OS/2 BMP
  WMF 15    Windows Meta File
  EPS 16    Encapsulated Post Script
  ```

```
TIFLZW 17     TIF Format with LZW compression
LEAD 20       LEAD Proprietary
LEAD1JFIF 21  JPEG 4:1:1
LEAD1JTIF 22  JPEG 4:1:1
LEAD2JFIF 23  JPEG 4:2:2
LEAD2JTIF 24  JPEG 4:2:2
CCITT 25      TIFF CCITT
LEAD1BIT 26   LEAD 1 bit, lossless compression
CCITT_GROUP3_1DIM 27    CCITT Group3 one dimension
CCITT_GROUP3_2DIM 28    CCITT Group3 two dimensions
CCITT_GROUP4 29   CCITT Group4 two dimensions
LEAD_NOLOSS 30    LEAD Proprietary Lossless
LEAD1BITA 31      old LEAD 1 bit, lossless compression
CALS 50   CALS
MAC 51    MAC
IMG 52    IMG
MSP 53    MSP
WPG 54    WPG
RAS 55    RAS
PCT 56    PCT
PCD 57    PCD
DXF 58    DXF
AVI 59    AVI
WAV 60    WAV
FLI 61    FLI
CGM 62    CGM
EPSTIFF 63    EPS with TIFF Preview
EPSWMF 64     EPS with Metafile Preview
CMPNOLOSS 65  CMPNOLOSS
FAX_G3_1D 66  FAX_G3_1D
FAX_G3_2D 67  FAX_G3_2D
FAX_G4 68     FAX_G4
WFX_G3_1D 69  WFX_G3_1D
WFX_G4 70     WFX_G4
ICA_G3_1D 71  ICA_G3_1D
ICA_G3_2D 72  ICA_G3_2D
ICA_G4 73     ICA_G4
OS2_2 74      OS2_2
PNG 75        PNG
PSD 76        PSD
RAWICA_G3_1D 77      RAWICA_G3_1D
RAWICA_G3_2D 78      RAWICA_G3_2D
RAWICA_G4 79         RAWICA_G4
FPX 80               FlashPix, no compression
FPX_SINGLE_COLOR 81  FlashPix, compression 'single color' method
FPX_JPEG 82      FlashPix, compression JPEG
FPX_JPEG_QFACTOR 83    FlashPix, compression JPEG, specify qFactor
BMP_RLE 84     compressed Windows BMP
TIF_CMYK 85    TIFF no compression, CMYK data
TIFLZW_CMYK 86 TIFF LZW compression, CMYK data
TIF_PACKBITS 87       TIFF PackBits compression, RGB data
TIF_PACKBITS_CMYK 88  TIFF PackBits compression, CMYK data
DICOM_GRAY 89     DICOM_GRAY
DICOM_COLOR 90    DICOM_COLOR
WIN_ICO 91        WIN_ICO
WIN_CUR 92        WIN_CUR
TIF_YCC 93        TIFF YcbCr color space, no compression
TIFLZW_YCC 94          TIFF YcbCr color space, LZW compression
TIF_PACKBITS_YCC 95    TIFF YcbCr color space, PackBits compression
EXIF 96       uncompressed RGB Exif file
EXIF_YCC 97   uncompressed YcbCr Exif file
EXIF_JPEG 98  JPEG compressed Exif file
AWD 99        Microsoft Fax format
FASTEST 100   for ISIS only! use the data as is, from the ISIS Scanner
```

```
Default       CCITT_GROUP4
```

depth

is the number of bits per pixel of the clipped image. Default: 1

resolution

is the assumed resolution of the image. All coordinates are supposed to be in this resolution. If the real resolution of the image is different, then a conversion takes place. If resolution=0 then no conversion takes place. Default: 0

left,top,right,bottom are 1-based

processing:

1. At first clipping is performed using left,top,right,bottom.

2. If seek is defined then the program seeks a vertical (left,right) or horizontal (top,bottom) line in the clipped rectangle from the edge of the clipped rectangle seek pixels in direction to the middle of the rectangle.

3. If a line was found then a new clipping position is defined at the position of this line plus (or minus) indent (a plus means always the direction to the middle of the rectangle).

4. If length is defined then the new clipped rectangle will get a width (or height) of length.

5. If at least one line was found a second clipping is performed.

CLIP creates the following hashtable entries if the image could be read, where <column> is the name of the column (or its alias) where CLIP is specified:

<column>.RC

0 ok

1 error from L_CopyBitmap

2 error from L_LoadBitmapMemory

3 error from L_CopyBitmapRect

4 error from L_SaveBitmapMemory

5 error from appCompress

6 error from L_RotateBitmap

7 error from L_SharpenBitmap

8 error from L_ColorResBitmap

-1 exception has occured

<column>.LEFT

is the left edge of the rectangle

<column>.TOP

is the the upper edge of the rectangle

<column>.RIGHT

the right edge of the rectangle

<column>.BOTTOM

is the lower edge of the rectangle

<column>.WIDTH

is the width

<column>.HEIGHT

is the height

`<column>.BPP`
are the Bits Per Pixel
`<column>.XRES`
is the X-Resolution
`<column>.YRES`
is the Y-Resolution
`<column>.FORMAT`
is the image format (integer, see above)
`<column>.FORMATSTRING`
is the image format (String, see above)

- **CMPEXT**
  Return code: index of the key that differs or 0 if all keys match
  Condition: true if the Hashtable sets are equal, otherwise false
  Description:
  Performs a compare of 2 sets of Hashtable keys. The input is ignored and remains unchanged.
  Syntax
  `CMPEXT[count[,fid1[,val1[,fid2[,val2]]]]]`
  where:
  `count`
  Count of new extensions. Default: 0
  `fid1`
  Stem name of the new field ID. Default: empty
  `val1`
  Stem name of the new value. Default: empty
  `fid2`
  Stem name of the old field ID. Default: empty
  `val2`
  Stem name of the old value. Default: empty

- **COUNTPIXELS**
  Return code:
  0 - o.k.
  1 - error from L_GetBitmapHistogram
  2 - error from L_LoadBitmapMemory
  3 - no mono bitmap
  4 - error from L_SaveBitmapMemory
  Condition: unchanged
  Description:
  Counts the black and white pixels of an image.
  Syntax
  `COUNTPIXELS"basename"`
  where
  `basename`
  is the base name for following keys created in the hashtable.
  Default: PIXELS
  PIXELS.0 8
  PIXELS.1 black pixels in percent
  PIXELS.2 white pixels in percent
  PIXELS.3 black pixels
  PIXELS.4 white pixels
  PIXELS.5 image width
  PIXELS.6 image height
  PIXELS.7 LeadTools rc
  PIXELS.8 image size in pixels

- **CSL**
  Return code: unchanged
  Condition: unchanged
  Description:
  Returns the number of remaining licenses for signatures or:
  -3 the query for signatures in the database failed
  -2 splm2 or licences not available
  -1 no (more) signatures are allowed
  0 no limit or not checked
  Syntax
  `CSL[max[,used]]`
  where:
  `max`
  is the name of the hashtable key where the number of licenses is stored. Default: SIGNATURE.MAX
  `used`
  is the name of the hashtable key where the number of currently active signatures is stored. Default: SIGNATURE.USED

- **GRAYCLEANZIP**

  Return code:

  0 - on success

  1 - on error

  2 - cleaning failed

  3 - too few non-white pixels

  4 - image too big

  Condition: unchanged

  Description:

  Converts a grayscale Tiff Image to Windows bitmap, cleans this bitmap using SIC_Proper and compresses it using zlib

  Syntax

  `GRAYCLEANZIP[clean-level[,compression[,minPixels[,maxSize]]]]`

  where:

  `clean-level`

  is the level of cleaning (hexadecimal):

  00000ii

  ii=index of proper fine tuning parameter set

  0000100 NOISE

  0000200 BACKGROUND

  0000400 TEXTURE

  0000800 LINE

  0001000 STAMP

  0002000 RECONSTRUCT

  0004000 CLEANFRAME

  0010000 NONE

  0020000 HOMOGEN

  4000000 PSEUDOCOLOR

  8000000 AUTOMATIC

  If no bit is set, no cleaning takes place. The levels can be combined by adding their values (hexadecimal). Default: 8022000

  `compression`

  is the type of compression:

  C - the result is a Bitmap, zlib compressed

  U - the result is a Tiff image, uncompressed.

  Default: C

  `minPixels`

  The minimum amount of non-white pixels. If the amount of non-white pixels is less than minPixels, null is returned. Default: 0, i.e. no pixel check takes place

  `maxSize`

The maximum size of an image in bytes. If the maximum size is greater than maxSize, null is returned. Default: 0, i.e. no size check takes place

GRAYCLEANZIP creates the following hashtable keys if the image could be read, where <column> is the name of the column (or its alias) where GRAYCLEANZIP is specified:

`<column>.RC`

0 on success, otherwise 1

`<column>.WIDTH`

Width

`<column>.HEIGHT`

Height

`<column>.BPP`

Bits Per Pixel

`<column>.XRES`

X-Resolution (integer)

`<column>.YRES`

Y-Resolution (integer)

- **IFD**

  Return code: unchanged

  Condition: unchanged

  Description:

  Returns a Tiff Image from the input (a Multi Tiff byte array).

  Syntax

  `IFD[page-number[,flags]]`

  where:

  `page-number`

  The page number inside the Multi Tiff File. Default: 1

  page-number is 1-based.

  `flags`

  0 - no values of additional IFDs are stored

  1 - if an additional IFD of the type Unisys IXPS, WEIRD or ISIS IFD exists in the specified page, their values are stored in the hashtable with the key name TAG.<tag-number>.

  2 - all tags of the current IFD are stored with the name TAG.<tag-name>. The following names are known:

```
SUBFILETYPE                 (Tag 254)
OSUBFILETYPE                (Tag 255)
IMAGEWIDTH                  (Tag 256)
IMAGELENGTH                 (Tag 257)
BITSPERSAMPLE               (Tag 258)
COMPRESSION                 (Tag 259)
PHOTOMETRIC                 (Tag 262)
THRESHHOLDING               (Tag 263)
CELLWIDTH                   (Tag 264)
CELLLENGTH                  (Tag 265)
FILLORDER                   (Tag 266)
DOCUMENTNAME                (Tag 269)
```

```
IMAGEDESCRIPTION            (Tag 270)
MAKE                        (Tag 271)
MODEL                       (Tag 272)
STRIPOFFSETS                (Tag 273)
ORIENTATION                 (Tag 274)
SAMPLESPERPIXEL             (Tag 277)
ROWSPERSTRIP                (Tag 278)
STRIPBYTECOUNTS             (Tag 279)
MINSAMPLEVALUE              (Tag 280)
MAXSAMPLEVALUE              (Tag 281)
XRESOLUTION                 (Tag 282)
YRESOLUTION                 (Tag 283)
PLANARCONFIG                (Tag 284)
PAGENAME                    (Tag 285)
XPOSITION                   (Tag 286)
YPOSITION                   (Tag 287)
FREEOFFSETS                 (Tag 288)
FREEBYTECOUNTS              (Tag 289)
GRAYRESPONSEUNIT            (Tag 290)
GRAYRESPONSECURVE           (Tag 291)
GROUP3OPTIONS               (Tag 292)
GROUP4OPTIONS               (Tag 293)
RESOLUTIONUNIT              (Tag 296)
PAGENUMBER                  (Tag 297)
COLORRESPONSEUNIT           (Tag 300)
TRANSFERFUNCTION            (Tag 301)
SOFTWARE                    (Tag 305)
DATETIME                    (Tag 306)
ARTIST                      (Tag 315)
HOSTCOMPUTER                (Tag 316)
PREDICTOR                   (Tag 317)
WHITEPOINT                  (Tag 318)
PRIMARYCHROMATICITIES       (Tag 319)
COLORMAP                    (Tag 320)
HALFTONEHINTS               (Tag 321)
TILEWIDTH                   (Tag 322)
TILELENGTH                  (Tag 323)
TILEOFFSETS                 (Tag 324)
TILEBYTECOUNTS              (Tag 325)
BADFAXLINES                 (Tag 326)
CLEANFAXDATA                (Tag 327)
CONSECUTIVEBADFAXLINES      (Tag 328)
INKSET                      (Tag 332)
INKNAMES                    (Tag 333)
DOTRANGE                    (Tag 336)
TARGETPRINTER               (Tag 337)
EXTRASAMPLES                (Tag 338)
SAMPLEFORMAT                (Tag 339)
SMINSAMPLEVALUE             (Tag 340)
SMAXSAMPLEVALUE             (Tag 341)
JPEGPROC                    (Tag 512)
JPEGIFOFFSET                (Tag 513)
JPEGIFBYTECOUNT             (Tag 514)
JPEGRESTARTINTERVAL         (Tag 515)
JPEGLOSSLESSPREDICTORS      (Tag 517)
JPEGPOINTTRANSFORM          (Tag 518)
JPEGQTABLES                 (Tag 519)
JPEGDCTABLES                (Tag 520)
JPEGACTABLES                (Tag 521)
YCBCRCOEFFICIENTS           (Tag 529)
YCBCRSUBSAMPLING            (Tag 530)
YCBCRPOSITIONING            (Tag 531)
REFERENCEBLACKWHITE         (Tag 532)
REFPTS                      (Tag 32953)
```

```
REGIONTACKPOINT             (Tag 32954)
REGIONWARPCORNERS           (Tag 32955)
REGIONAFFINE                (Tag 32956)
MATTEING                    (Tag 32995)
DATATYPE                    (Tag 32996)
IMAGEDEPTH                  (Tag 32997)
TILEDEPTH                   (Tag 32998)
PIXAR_IMAGEFULLWIDTH        (Tag 33300)
PIXAR_IMAGEFULLLENGTH       (Tag 33301)
WRITERSERIALNUMBER          (Tag 33405)
COPYRIGHT                   (Tag 33432)
IT8SITE                     (Tag 34016)
IT8COLORSEQUENCE            (Tag 34017)
IT8HEADER                   (Tag 34018)
IT8RASTERPADDING            (Tag 34019)
IT8BITSPERRUNLENGTH         (Tag 34020)
IT8BITSPEREXTENDEDRUNLENGTH (Tag 34021)
IT8COLORTABLE               (Tag 34022)
IT8IMAGECOLORINDICATOR      (Tag 34023)
IT8BKGCOLORINDICATOR        (Tag 34024)
IT8IMAGECOLORVALUE          (Tag 34025)
IT8BKGCOLORVALUE            (Tag 34026)
IT8PIXELINTENSITYRANGE      (Tag 34027)
IT8TRANSPARENCYINDICATOR    (Tag 34028)
IT8COLORCHARACTERIZATION    (Tag 34029)
FRAMECOUNT                  (Tag 34232)
ICCPROFILE                  (Tag 34675)
JBIGOPTIONS                 (Tag 34750)
FAXRECVPARAMS               (Tag 34908)
FAXSUBADDRESS               (Tag 34909)
FAXRECVTIME                 (Tag 34910)
DCSHUESHIFTVALUES           (Tag 65535)
UNISYS_ISIS_IFD             (Tag 33881)
UNISYS_SIDE                 (Tag 33882)
UNISYS_IXPS_IFD             (Tag 33884)
BANCTEC_IFD                 (Tag 34975)
```

All other tags are stored with their number as name. Default: 1

IFD creates the following hashtable keys if the image could be read, where <column> is the name of the column (or its alias) where IFD is specified:

`<column>.RC`

0

`<column>.WIDTH`

Width

`<column>.HEIGHT`

Height

`<column>.BPP`

Bits Per Pixel

`<column>.XRES`

X-Resolution (integer)

`<column>.YRES`

Y-Resolution (integer)

`<column>.PAGES`

Number of Pages

- **LINESEARCH**
  Return code: unchanged
  Condition: unchanged
  Description:
  Provides either rectangle-coordinates of an image, that contain handwriting, or a changed image.
  Syntax
  `LINESEARCH"[search-properties-file]"`
  where:
  `search-properties-file`
  contains all settings for the search. Default: de.softpro.signplus.service.LineSearch
  The following keys are supported:
  `search.crop`
  rectangle of the image, where the search shall take place („<left>,<top>,<right>,<bottom>“). Default: „1,1,-1,-1“ (the whole image)
  `framesName`
  basename for storing the found rectangles/lines
  `control`
  defines bitwise the activities:
  1 - return an image (otherwise frames are returned)
  2 - search for horizontal lines
  4 - remove lines
  8 - search for handwritings
  16 - search for vertical lines
  32 - don't merge the frames found
  `resolution`
  resolution in dpi for pixel-related parameters
  `line.minLength`
  minimum length of a simple line in pixel
  `line.minLengthComposed`
  minimum length of a composed line in pixel
  `line.maxGap`
  maximum count of consecutive white pixels inside a simple line
  `line.removeX`
  count of pixels right and left of a point on a line that are involved to calculate the pixels that can be deleted
  `line.removeY`
  count of pixels beyond and beneath a point on a line that are involved to calculate the pixels that can be deleted
  `signature.minWidth`
  minimum width in pixel of a rectangle, that contains contiguous black pixels, to be part of a handwriting
  `signature.minHeight`

minimum height in pixel of a rectangle, that contains contiguous black pixels, to be part of a handwriting

`signature.maxWidth`

maximum width in pixel of a rectangle, that contains contiguous black pixels, to be part of a handwriting

`signature.maxHeight`

maximum height in pixel of a rectangle, that contains contiguous black pixels, to be part of a handwriting

`signature.minBlackPixels`

minimum count of black pixels for a rectangle to be part of a handwriting (currently not used)

`signature.maxBlackPixels`

maximum count of black pixels for a rectangle to be part of a handwriting (currently not used)

`signature.minSquarePixels`

minimum count of pixels (width * height) for a composed rectangle

`signature.maxSquarePixels`

maximum count of pixels (width * height) for a composed rectangle

`signature.maxGapX`

maximum horizontal gap between two simple rectangles to be part of the same composed rectangle

`signature.maxGapY`

maximum vertical gap between two simple rectangles to be part of the same composed rectangle

`signature.minWidthComposed`

maximum width in pixel of a rectangle containing handwriting

`signature.minHeightComposed`

maximum height in pixel of a rectangle containing handwriting

If rectangle-coordinates are returned, they will be stored in the hashtable with the defined basename, where:

`<name>.0`

count of rectangles/lines

`<name>.1`

is the first rectangle in the format „<left>,<top>,<right>,<bottom>“

`<name>.2`

is the second rectangle etc.

- **LTIF**

  Return code: unchanged

  Condition: unchanged

  Description:

  Returns a Tiff Image from the input,a Multi Tiff File.

  Syntax

  `LTIF[page-number[,flags]]`

  where:

  `page-number`

  is the page number inside the Multi Tiff File. page-number is 1-based. Default: 1

  `flags`

  0 - no values of additional IFDs are stored

  1 - if an additional IFD of the type Unisys IXPS, WEIRD or ISIS IFD exists in the specified page, their values are stored in the hashtable with the key name TAG.<tag-number>.

  2 - all tags of the current IFD are stored with the name TAG.<tag-name>. The following names are known:

```
SUBFILETYPE                 (Tag 254)
OSUBFILETYPE                (Tag 255)
IMAGEWIDTH                  (Tag 256)
IMAGELENGTH                 (Tag 257)
BITSPERSAMPLE               (Tag 258)
COMPRESSION                 (Tag 259)
PHOTOMETRIC                 (Tag 262)
THRESHHOLDING               (Tag 263)
CELLWIDTH                   (Tag 264)
CELLLENGTH                  (Tag 265)
FILLORDER                   (Tag 266)
DOCUMENTNAME                (Tag 269)
IMAGEDESCRIPTION            (Tag 270)
MAKE                        (Tag 271)
MODEL                       (Tag 272)
STRIPOFFSETS                (Tag 273)
ORIENTATION                 (Tag 274)
SAMPLESPERPIXEL             (Tag 277)
ROWSPERSTRIP                (Tag 278)
STRIPBYTECOUNTS             (Tag 279)
MINSAMPLEVALUE              (Tag 280)
MAXSAMPLEVALUE              (Tag 281)
XRESOLUTION                 (Tag 282)
YRESOLUTION                 (Tag 283)
PLANARCONFIG                (Tag 284)
PAGENAME                    (Tag 285)
XPOSITION                   (Tag 286)
YPOSITION                   (Tag 287)
FREEOFFSETS                 (Tag 288)
FREEBYTECOUNTS              (Tag 289)
GRAYRESPONSEUNIT            (Tag 290)
GRAYRESPONSECURVE           (Tag 291)
GROUP3OPTIONS               (Tag 292)
GROUP4OPTIONS               (Tag 293)
RESOLUTIONUNIT              (Tag 296)
PAGENUMBER                  (Tag 297)
COLORRESPONSEUNIT           (Tag 300)
TRANSFERFUNCTION            (Tag 301)
SOFTWARE                    (Tag 305)
```

```
DATETIME                    (Tag 306)
ARTIST                      (Tag 315)
HOSTCOMPUTER                (Tag 316)
PREDICTOR                   (Tag 317)
WHITEPOINT                  (Tag 318)
PRIMARYCHROMATICITIES       (Tag 319)
COLORMAP                    (Tag 320)
HALFTONEHINTS               (Tag 321)
TILEWIDTH                   (Tag 322)
TILELENGTH                  (Tag 323)
TILEOFFSETS                 (Tag 324)
TILEBYTECOUNTS              (Tag 325)
BADFAXLINES                 (Tag 326)
CLEANFAXDATA                (Tag 327)
CONSECUTIVEBADFAXLINES      (Tag 328)
INKSET                      (Tag 332)
INKNAMES                    (Tag 333)
DOTRANGE                    (Tag 336)
TARGETPRINTER               (Tag 337)
EXTRASAMPLES                (Tag 338)
SAMPLEFORMAT                (Tag 339)
SMINSAMPLEVALUE             (Tag 340)
SMAXSAMPLEVALUE             (Tag 341)
JPEGPROC                    (Tag 512)
JPEGIFOFFSET                (Tag 513)
JPEGIFBYTECOUNT             (Tag 514)
JPEGRESTARTINTERVAL         (Tag 515)
JPEGLOSSLESSPREDICTORS      (Tag 517)
JPEGPOINTTRANSFORM          (Tag 518)
JPEGQTABLES                 (Tag 519)
JPEGDCTABLES                (Tag 520)
JPEGACTABLES                (Tag 521)
YCBCRCOEFFICIENTS           (Tag 529)
YCBCRSUBSAMPLING            (Tag 530)
YCBCRPOSITIONING            (Tag 531)
REFERENCEBLACKWHITE         (Tag 532)
REFPTS                      (Tag 32953)
REGIONTACKPOINT             (Tag 32954)
REGIONWARPCORNERS           (Tag 32955)
REGIONAFFINE                (Tag 32956)
MATTEING                    (Tag 32995)
DATATYPE                    (Tag 32996)
IMAGEDEPTH                  (Tag 32997)
TILEDEPTH                   (Tag 32998)
PIXAR_IMAGEFULLWIDTH        (Tag 33300)
PIXAR_IMAGEFULLLENGTH       (Tag 33301)
WRITERSERIALNUMBER          (Tag 33405)
COPYRIGHT                   (Tag 33432)
IT8SITE                     (Tag 34016)
IT8COLORSEQUENCE            (Tag 34017)
IT8HEADER                   (Tag 34018)
IT8RASTERPADDING            (Tag 34019)
IT8BITSPERRUNLENGTH         (Tag 34020)
IT8BITSPEREXTENDEDRUNLENGTH (Tag 34021)
IT8COLORTABLE               (Tag 34022)
IT8IMAGECOLORINDICATOR      (Tag 34023)
IT8BKGCOLORINDICATOR        (Tag 34024)
IT8IMAGECOLORVALUE          (Tag 34025)
IT8BKGCOLORVALUE            (Tag 34026)
IT8PIXELINTENSITYRANGE      (Tag 34027)
IT8TRANSPARENCYINDICATOR    (Tag 34028)
IT8COLORCHARACTERIZATION    (Tag 34029)
FRAMECOUNT                  (Tag 34232)
ICCPROFILE                  (Tag 34675)
```

```
JBIGOPTIONS                 (Tag 34750)
FAXRECVPARAMS               (Tag 34908)
FAXSUBADDRESS               (Tag 34909)
FAXRECVTIME                 (Tag 34910)
DCSHUESHIFTVALUES           (Tag 65535)
UNISYS_ISIS_IFD             (Tag 33881)
UNISYS_SIDE                 (Tag 33882)
UNISYS_IXPS_IFD             (Tag 33884)
BANCTEC_IFD                 (Tag 34975)
```

All other tags are stored with their number as name. Default: 1

LTIF creates the following hashtable keys if the image could be read, where <column> is the name of the column (or its alias) where LTIF is specified:

`<column>.RC`

0

`<column>.WIDTH`

Width

`<column>.HEIGHT`

Height

`<column>.BPP`

Bits Per Pixel

`<column>.XRES`

X-Resolution (integer)

`<column>.YRES`

Y-Resolution (integer)

`<column>.PAGES`

Number of Pages

- **MONOCLEAN**

  Return code:

  0 on success

  -1 otherwise

  Condition: unchanged

  Description:

  Cleans a mono Image (Tiff format) using Sival's FsxClean.

  Syntax

  `MONOCLEAN[level[,lines[,basename]]]`

  where:

  `level`

  The cleaning level. The lowest level is 0 (cleans nothing), the highest level is 1000 (cleans all). Default: empty, i.e. no frame-cleaning, but normal cleaning

  `lines`

  shall lines be removed? 0 – no, 1 - yes

  Default: empty, i.e. no frame-cleaning, but normal cleaning

  If one of the parameters level and lines is not numeric and not empty, no cleaning takes place, but the frames calculated by sival are stored in variables of the hashtable:

  <level>.0 - the number of frames

  <level>.1 - the coordinates of the first frame in pixels (x-left,y-upper,x-right,y-lower)

  <level>.2 - the coordinates of the second frame etc.

  In another variable of this hashtable with the name of lines are written the classifications of these frames (0 – frame does not belong to the signature, 999999 – Frame belongs definitely to the signature):

  `<lines>.0`

  the number of frames

  `<lines>.1`

  the classification of the first frame etc.

  `basename`

  The base name for some resulting hashtable keys. Default: the name of the column (or its alias) where MONOCLEAN is specified

  `<basename>.RC`

  the returncode of Sival, 0 is o.k.

  `<basename>.WIDTH`

  Width

  `<basename>.HEIGHT`

  Height

  `<basename>.LEFT`

  the left edge of the signature snippet

  `<basename>.TOP`

  the upper edge of the signature snippet

  `<basename>.RIGHT`

the right edge of the signature snippet

`<basename>.BOTTOM`

the lower edge of the signature snippet

- **PAD**
  Return code: PAD matchrate or -1 on error
  Condition: true if the check is a PAD, otherwise false
  Description:
  Performs a PAD (pre-authorized draft) check.
  Syntax
  `PAD[padlevel[,accountholderlevel[,blacklistlevel[,result[,accountholder[,IRD]]]]]]`
  where:
  `padlevel`
  Minimum confidence level PAD, a number between 0 and 100. Default: 0
  `accountholderlevel`
  Minimum confidence level Account Holder Name, a number between 0 and 100. Default: 0
  `blacklistlevel`
  Minimum confidence level Blacklist, a number between 0 and 100. Default: 0
  result
  Base name of the result key. Default: empty
  The following keys will be defined:
  `<result>.0`
  8
  `<result>.1`
  1 - PAD, 0 - no PAD
  `<result>.2`
  PAD matchrate
  `<result>.3`
  1 - Account Holder found, 0 - not found
  `<result>.4`
  Account Holder matchrate
  `<result>.5`
  1 - Blacklist entry found, 0 - not found
  `<result>.6`
  Blacklist entry matchrate
  `<result>.7`
  Blacklist entry name if Blacklist entry found
  `<result>.8`
  Matching PAD keyword if PAD detected
  `accountholder`
  Name of the hashtable key containing the Account Holder as a result of this function. Default: empty
  `IRD`
  !=0 - item is an IRD; otherwise not
- **SBSTAT**
  Return code: unchanged

Condition: unchanged

Description:

Updates the SignBase Statistics table SB_STATISTICS. The following keys from the hashtable are used and must be filled:

- BNO

- TIMESTAMP

Syntax

```
SBSTAT[statisticId[,count[,commit]]]
```

where:

`statisticId`

the statistic Id; a positive number

`count`

Count to be added to column VALUE of the row with this StatisticId. Default: 1

`commit`

1 - force commit

0 - do not force commit

Default: 0

Latest at the end of the service program all changes are committed.

- **SELECT**

  Return code: unchanged

  Condition: unchanged

  Description:

  A SELECT is performed on the database table defined in the resource file in parameter 2 of the constructor.

  The input contains the name of the file where the result of the select is stored. The format of this file is also defined in the resource file in parameter 2 of the constructor.

  Syntax

  ```
  SELECT
  ```

- **SIMPLICITY**

  Return code:

  the simplicity of the signature from 0 to 100 (100 means most simple) or -1 on error

  Condition: unchanged

  Description:

  Compares 1 signature with a set of references. The input is considered to contain the signature, a byte array. The format must be tiff monochrome or bitmap.

  Syntax

  `SIVALCOMPARE[matchrate[,simplicity[,control[,reference[,width[,height[,xres[,yres[,matchratekey[,simplicitykey]]]]]]]]]]`

  where:

  `matchrate`

  minimum matchrate

  `simplicity`

  maximum simplicity

  `control`

  0 - compare until the matchrate has been reached

  1 - compare against all references

  2 - parameters reference, width, height, xres and yres are not the basenames, but the the complete names, i.e. there is only one reference

  `reference`

  Basename of the references, default is "SIC_M". The format for the references must be the proprietary SignBase reference format. The count of references is expected in <reference>.0

  `width`

  basename of the widths. Default: "WIDTH"

  `height`

  basename of the heights. Default: "HEIGHT"

  `xres`

  basename of the x-resolutions. Default: "X_RES"

  `yres`

  basename of the y-resolutions. Default: "Y_RES"

  `matchratekey`

  Basename for the matchrates, default is empty. If not empty, the matchrate of the corresponding reference is stored in <matchratekey>.1 for the first reference etc. <matchratekey>.0 contains the count of references.

  `simplicitykey`

  Basename for the simplicities, default is empty. If not empty, the simplicity of the corresponding reference is stored in <simplicitykey>.1 for the first reference etc. <simplicitykey>.0 contains the count of references.

- **SIZETYPE**

  Return code: unchanged

  Condition: unchanged

  Description:

  Returns the type of the first range where width and height are inside or the default type if none of the ranges matches.

  Syntax

  `SIZETYPE[width[,height[,xres[,yres]]]]`

  where:

  `width`

  The width (integer) or the name of the hashtable key containing the width. Default: "TAG.IMAGEWIDTH"

  `height`

  The height (integer) or the name of the hashtable key containing the heights. Default: "TAG.IMAGELENGTH"

  `xres`

  The X-resolution (integer) or the name of the hashtable key containing the X-resolution. Default: "TAG.XRESOLUTION"

  `yres`

  The Y-resolution (integer) or the name of the hashtable key containing the Y-resolution. Default: "TAG.YRESOLUTION"

- **TEMPLATE**

  Return code: unchanged

  Condition: unchanged

  Description:

  Calculates a template byte array for one or more images.

  Syntax

  `TEMPLATE[basename]`

  where:

  `basename`

  Basename of all keys containing the images that are supposed to be used to build a template. <basename>.0 contains the count of images to be used, <basename>.1 contains the 1st image, <basename>.2 contains the 2nd image etc. If <basename>.0 is not defined or empty, it is assumed that there is only 1 image, stored in <basename>.

- **UNZIPP**

  Return code: unchanged

  Condition: unchanged

  Description:

  Unzips the input using the unzip function in spjdec.dll. If the unzipping fails, the input value is returned.

  Syntax

  `UNZIPP`

- **VARIANTCOMPARE**

  Return code:

  the count of similar variants or -1 on error

  Condition: true if the variant compare succeeded, otherwise false

  Description:

  Compares a set of variants with each other. The input is ignored.

  Syntax

  `VARIANTCOMPARE[matchrate[,reference[,width[,height[,xres[,yres[,signatureOID[,signator`
  `[,signo[,type[,result]]]]]]]]]]`

  where:

  `matchrate`

  Minimum matchrate

  `reference`

  Basename of the references, default is "SIC_M". The format for the references must be the proprietary SignBase reference format. The count of references is expected in <reference>.0

  `width`

  Basename of the widths. Default: "WIDTH"

  `height`

  Basename of the heights. Default: "HEIGHT"

  `xres`

  Basename of the x-resolutions. Default: "X_RES"

  `yres`

  Basename of the y-resolutions. Default: "Y_RES"

  `signatureOID`

  Basename of the object Id of the signature. Default: "MOID"

  `signatoryOID`

  Basename of the object Id of the signatory. Default: "OID"

  `signo`

  Basename of the signo of the signatory. Default: "SIGNO"

  `type`

  Basename of the type of the signatory. Default: "TYPE"

  `result`

  Basename for the resulting object IDs that are similar to other variants, default is "RESULT.OID". The count of similar variants is stored in <result>.0.

Trace levels

All service programs have a log area, where logging information are written to. It is also possible to write the log to a file. The following trace levels are defined:

Name	Value	Description
ERROR	1	Error-Messages

Name	Value	Description
WARNING	2	Warnings
DEBUG	4	Debug Messages
INFO	8	Information
RESOURCE	16	List contents of the property files.
SQL	32	SQL activities, this level has no effect for service programs that do not work directly with a SQL database, like DataViewer, SignatureReferenceFilter, ResultLoader, ResultWriter.
SUBSTITUTE	64	Step by step protocol of the substitution process, this can be considered as a deeper debug trace level.
SELECTION	128	Step by step protocol of the selection process.
PERFORMANCE	256	Performance information. It is recommended to switch off as many other trace levels as possible, because they have an influence on the performance. Logs with trace level are not written during processing, only when stopping the program. It is important, to have this trace level switched on when stopping.

Work file processing

When working with service programs, that process input files (AccountLoader, SRF, Getter; ImageLoader, F3Loader, SRF), is it possible to define work files. This means, that for every data file, that was opened, a work file is created, that will contain information about the progress of processing. This has the following advantages:

- More than one service program can work on the same data directory without processing one data file twice.
- If a service program was stopped or crashed, the data file processing can continue later at the last position of processing.

Preconditions for the use of work files are:

- The data directory must be writable or a writable work directory is defined, common for all programs.
- The data files, that have been processed, must be renamed or deleted resp.

After complete processing an input file, the associated work file is deleted. After program stop or crash the work file persists to give information to later activated programs. This means that work files should not be changed or deleted manually.

Flow control

With some keys in the hashtable it is possible to modify the execution of the tableResource<n> tables. These keys are always deleted before processing the next input line, i.e. their settings are only valid for the current line and the current table resource file:

CONTROL.ACTION

Determines, which action will be applied for the current table:

I - INSERT

U - UPDATE

D - DELETE

S - SELECT

N - NONE: no database action takes place for this table

C - CONTINUE: no database action takes place for this table and for all subsequent tables for the current line

E - ERROR: processing of the current data file is stopped. This file will be marked as erroneous. Continue with the next data file.

Q - QUIT: same as ERROR, but the service program stops.

W - wait with processing the current input file, continue with the next file (only Getter)

This setting is valid only for the current table. After processing this table the key CONTROL.ACTION is reset. The following tables are processed according to the key action (in case of AccountLoader) or with an INSERT (in case of Getter), as long as the key CONTROL.ACTION is not defined again. If the resulting action is SELECT, the result of this query is stored in the hashtable:

`<alias>.0` number of selected rows

`<alias>` content of the first row

`<alias>.1` content of the first row

`<alias>.2` content of the 2. row etc.

If the key CONTROL.ACTION contains after the S a pattern „FILE=<filename>“, then the following text is interpreted as a filename and the result is not inserted into the hashtable, but written to this file.

The following table shows, for which service programs which values of CONTROL.ACTION are implemented:

ACTION	I	U	D	S	N	C	E	Q
Programm								
AccountLoader	+	+	+	+	+	+	+	+
ImageLoader	+	+	+	+	+	+	+	+
FraudFeedbackFileLoader	+	+	+	+	+	+	+	+
Getter	+	-	-	+	+	+	+	-
Putter	-	-	-	-	-	-	-	-
DFP	-	-	-	-	-	-	-	-
ResultLoader	-	-	-	-	-	-	-	-
ResultWriterLoader	-	-	-	-	-	-	-	
SignatureReferenceFilter	-	-	-	-	-	+	-	
DataViewer	-	-	-	-	-	-	-	

ACTION	I	U	D	S	N	C	E	Q
PasswordEncoder	-	-	-	-	-	-	-	
TableAccess	-	-	-	-	-	-	-	
XML-Loader	-	-	-	-	-	-	-	

CONTROL.SKIP

(only Getter and AccountLoader) defines, how many of the following tables are excluded from the processing of the current line.

Example

There are 5 tables defined, tableResource1 up to tableResource5.

During the processing of tableResource2 takes place the definition of CONTROL.SKIP=2. Then processing of tables is continued with tableResource5 and CONTROL.SKIP is set to 0. Hence a CONTROL.SKIP=0 has no effect. It is possible to use negative values for CONTROL.SKIP to achieve repeated execution of tables. Be careful with CONTROL.SKIP=-1(endless repetition of the same file). Also is it possible to use the name of the table that shall be executed next, instead of a number, in this case only the name of the resource file is used, not the path or package. If a resource file occurs more than once in the list of resources, the first index of the first found file is used.

CONTROL.ROWS

(only Getter and AccountLoader) the number of repetitions of processing the current resource file (Default: 1) before continuing with the next file. This value is calculated only for the first time of processing the file, a redefining of this value during the 2nd processing has no effect.

CONTROL.ROW

(only Getter and AccountLoader) the current repetition of the resource file, starting with 1. Setting the initial value and incrementing after processing is done automatically by the program. This feature is useful when processing stems of variables.

CONTROL.REJECT

(only AccountLoader) If set to 0 it suppresses the reject of lines when Insert (or Update) fails, because the record in the database already (or not) exists.

Reject file processing

When using AccountLoader or SignatureReferenceFilter, you have the possibility, records that could not be processed, to write to a so called reject file. The format of the records is not changed. Additional to this record at least one comment line is written to give information about the reason of the reject. To every data file a separate reject file is created, if any.

Example

```
# line 3:
CA002840    0000005710004846    KELLNER CRAIG   0000005710004846   USD    0
# COM.ibm.db2.jdbc.DB2Exception: [IBM][CLI Driver][DB2/NT] SQL0407N Assig…
```

Advantages

- Rejected records can be investigated better and are documented better than in a log
- By renaming of the reject file is it possible to re-process these files, after the reason of the failure, e.g. network problems, has been eliminated.

Reports

For AccountLoader and SignatureReferenceFilter it is possible to create report files. Per data line it can be created one (AccountLoader) or several (SignatureReferenceFilter) report entry. The format of those entries is defined in a properties file. All formulas defined there are resolved in the alphabetical order of its keys. Syntax for a formula:

```
${hashtable-key|<delm>< function1][<delm><function2>...]...}
```

where hashtable-key is the name of a key from the hashtable. The remaining parameters are described already in chapter Formulas.

The formulas ${...} can be nested and are resolved starting from the innermost ones.

The following keys are predefined by the program:

Key	Default	Description
REPORT		1 - if a report is to be created, otherwise 0
REPORT.DB		1 - if only database changes are to be reported 0 - all activities are reported
REPORT.DELIMITER	";"	Delimiting character for the fields of one report line

Additional to the keys above SignatureReferenceFilter define the following keys:

Key	Description
REPORT.BNO	BNO
REPORT.COUNTRYID	COUNTRYID
REPORT.BANKCODE	BANKCODE
REPORT.CUSTOMERNO	CUSTOMERNO
REPORT.ACCOUNT	ACCOUNT
REPORT.DOCREFNO	DOCREFNO

Key	Description
REPORT.ACTION	I - New reference inserted C - No reference inserted - account complete O - An old variant has been replaced Q - No reference inserted - bad quality R - No reference inserted - item image considered risky N - Not inserted S - Not selected for insert J - Rejected A - Amount too little T - Signature too similar to an existing one B - Missing SignBase data D - Item not inserted because it is an IRD U - No reference inserted - not assigned to a Signatory
REPORT.SIGNATORIES	Number of signatories
REPORT.VARIANTS	Number of variants
REPORT.SIGNO	SIGNO of the new variant
REPORT.BESTMATCHRATE	Best match rate against the already defined signatories
REPORT.SIGNO_BESTMATCH	SIGNO of the signatory with the best match rate
REPORT.QUALITY	00 - OK 01 - No image left after clipping white space 02 - Snippet height to low 03 - Snippet width to slim 04 - Too few pixels left in image 05 - Not enough parameters found in snippet 06 - Too much pixels left in image
REPORT.CUSTOMERTYPE	CUSTOMERTYPE
REPORT.ADD1	The size of a new signature in bytes

Furthermore any keys can be used that have been defined before.

Example

All records that have changed the database shall be reported in the file

```
c:\report\report.txt
```

in the format

```
<ISO-date of the data file>|<customer number>
```

In the hashtable the following keys are defined:

```
REPORT.DELIMITER=|
REPORT.PATH=c:\report\report.txt
REPORT.CUSTOMERNO=<customer number>
```

```
REPORT.CHG=1 - database has been changed, 0 - not changed
```

The configuration of the report properties file looked like this:

```
col01=${FILE.DATE|SUB1.10|FMT"%s${REPORT.DELIMITER}|SAVE"REPORT.LINE"}
col02=${REPORT.LINE|FMT"%s${CUSTOMERNO}"|SAVE"REPORT.LINE"}
col03=${REPORT.LINE|TEST"${REPORT.CHG}"==1?WRITEFILE"${REPORT.PATH}"}
```

AccountLoader

This service program conducts, as requested, a comparison of the customer, account or signatory data with a customer information system. This program can also be employed for initial recording of customer, account and signatory data in the SignBase database (initial load). The following features are possible and configurable:

- Creating, modifying and deleting (physical and logical) customers
- Creating, modifying and deleting (physical and logical) accounts
- Creating signatories
- Creating variants
- Creating rules
- Creating mono and gray signatures, inclusive cleaning
- Creating of Account Images
- Historizing of old entries when modifying or deleting
- Copying of a whole customer
- Update all signatories

After successfully processing a data file, this data file and it's activate file, so far specified, will be deleted or renamed. If the processing of a data file failed, the data file will be renamed to avoid a second processing and the log will be written to a file with the same name as the data file, but with a different extension.

Configuration of the AccountLoader

Additional to the above described configuration of all main configuration files there are the following keys:

Key	Default	Description
commitLine	true	true - every input record is committed to the database, otherwise not (but maybe via function COMMIT)

Key	Default	Description
action	"${LINE.CONTENT|SUB1.1}"	The action to be performed. This can be one of: C - Create data K - Delete data physically D - Delete data with historization H - Modify data with historization E - Create/Modify with historization U - Update all signatories X - Copy Customer
trailerRecord	false	If the data file has a trailing record, which is no regular data record, the contents of the last line is stored in the hashtable in key TRAILER.
activateSuffix(f)	value of dataSuffix	The suffix of an activate file with the same name as the data file. The existence of this file allows the processing of the data file.
customerlockTableResource.0	0	Count of table resource files for the customer locking mechanism
customerlockTableResource.0		1st table resource file for the customer locking mechanism
customerlockTableResource.2		2nd table resource file for the customer locking mechanism etc.
SBStatisticsTableResource		Table resource file to perform the SignBase statistics
protocolInsert	false	If true, then the count of inserts is written to the log when the program stops.
protocolUpdate	false	If true, then the count of updates is written to the log when the program stops.
protocolDelete	false	If true, then the count of deletes is written to the log when the program stops.
timestampFormat	yyyy-MM-dd HH:mm:ss.SSS	The format for time stamps in the SignBase statistics table

Special keys in the hashtable

The following keys are created automatically after processing of every table resource file:

Key	Description
LAST.RESULT	Number of rows changed by the last UPDATE, INSERT or DELETE.
LAST.TABLE	Name of the table of the last UPDATE, INSERT or DELETE.

Key	Description
TABLE	Name of the table of the last successful UPDATE, INSERT or DELETE.
REPORT.CHANGED	1 after a successful INSERT, UPDATE or DELETE statement, otherwise 0.
EXTRA.INFO	additional information appended to the log when the AccountLoader stops with a message like “I 2012-06-19 14:28:08.421 -: stopped after 8 records. elapsed time: 0:02:25.541”

This key is created after opening a data file:

Key	Description
TRAILER	The last record of the data file if the key trailerRecord is true, otherwise TRAILER is empty.

The following keys can be defined after reading a record from the data file and processing the first table properties file. They control the further processing:

Key	Action	Target	Default	Description
CONTROL.ACTION				N=NONE - do nothing C=CONTINUE - do nothing with the remaining tables E=ERROR - error in data file, process next data file S=SELECT - perform SQL SELECT I=INSERT - perform SQL INSERT D=DELETE - perform SQL DELETE U=UPDATE - perform SQL UPDATE
CONTROL.ROW			1	The current row (set automatically)
CONTROL.ROWS			1	The count of rows to be executed for a table resource
CONTROL.SKIP			empty	If numeric then skip relative to the current table resource, negative values are allowed, otherwise skip to the table resource with this name.
CONTROL.REJECT			1	0 - don't reject if an INSERT or UPDATE did not change any rows, otherwise reject all CONTROL.*-keys are deleted after processing a table resource
FILE.REPEATLINE			no	Repeat the current line (yes|no), this key is deleted after reading a line.

Key	Action	Target	Default	Description
ACTION				C - create M - modify K - delete physically D - delete with historization H - modify with historization E - create/modify U - update all(only for signatories) X - copy (only for a whole customer)
TARGET	all			C - customer A - account S - signatory V - variant I - account image
BNO	all	all		BNO
COUNTRYID	all	all		Country Id
BANKCODE	all	all		Bankcode
CUSTOMERNO	all	all		Customer number
SHORTNAME	CMHE	CA		Shortname for customer and account
VALUEDC	CMHE	C		Valued customer flag
CUSTOMERTYPE	CMHE	C		Customer type
ACCOUNT	all	AI		Account number
CURRENCY	CMHE	AS		Currency for account and rule
LNAME	CMHE	S		Last name
FNAME	CMHE	S		First name
TITLE	CMHE	S		Title
PERSONALID	CE	S		Personal Id
SIGLOC	CE	S		0 - no Image 1 - b/w-Image 2 - gray image R - re-use signature from db
IMAGEPATH	CE	SI		Image path and name
CUSTOMERSINCE	CMHE	C		Customer since (yyyyMMdd)
AI_ID	CHE	I		Image Id
AI_TEXT1	CHE	I		Image text 1
AI_TEXT2	CHE	I		Image text 2

Key	Action	Target	Default	Description
AI_STATUS	CHE	I		Image status: 0 - processed 1 - not processed
AI_SOURCE	CHE	I		Image source
AI_TYPE	CHE	I		Image type: 1 - b/w-Image 2 - gray image
AI_FREETEXT	CHE	I		Image free text
AI_PATH	CHE	I		Image path and name
Extensions *=1,...,EXTnTYPE.0, where n=last character of extension table name *TYPE.0 empty means, there are no extensions *TYPE.0=0 means, there are extensions, but no new values (for action H)				
EXT0FID.*	CMHE	C	*	FIELDIDs of Extension 0, if differing from the current number (*)
EXT0TYPE.0	CMHKE	C	empty	Number of customer extensions
EXT0TYPE.*	CMHE	C		CONTENTTYPEs of Extension 0
EXT0VALUE.*	CMHE	C		CONTENTs of Extension 0
EXT1FID.*	CMHE	A	*	FIELDIDs of Extension 1, if differing from the current number (*)
EXT1TYPE.0	CMHKE	CA		Number of account extensions
EXT1TYPE.*	CMHE	A		CONTENTTYPEs of Extension 1
EXT1VALUE.*	CMHE	A		CONTENTs of Extension 1
EXT2FID.*	CMHE	S	*	FIELDIDs of Extension 2, if differing from the current number (*)
EXT2TYPE.0	CMHKE	CS	empty	Number of signatory extensions
EXT2TYPE.*	CMHE	S		CONTENTTYPEs of Extension 2
EXT2VALUE.*	CMHE	S		CONTENTs of Extension 2
EXTBFID.*	CHE	I	*	FIELDIDs of Extension B, if differing from the current number (*)
EXTBTYPE.0	CHE	I	empty	Number of account image extensions
EXTBTYPE.*	CHE	I		CONTENTTYPEs of Extension B
EXTBVALUE.*	CHE	I		CONTENTs of Extension B
Rule Settings				

Key	Action	Target	Default	Description
RULE_CREATE	CHE	S	0	1 - create Rule 0 - no Rule
RULE_ACCOUNT	CHE	S	0	Rule Account
RULE_AMOUNT	CHE	S		Rule Amount
RULE_DELETE	HE	S	0	1 - delete Rule 0 - not
RULE_DELETE_ACCOUNT_ DEPENDENT	HE	S	0	1 - delete only Rules with the same ACCTNO 0 - delete all Rules
RULE_GROUPIN	CHE	S		Rule Groupin (instead of SIGNO), can be empty
RULE_POWER	CHE	S	S	Rule Power
RULE_LIMITFROM	CHE	S	empty	Rule Limit from (yyyyMMdd)
RULE_LIMITTO	CHE	S	empty	Rule Limit to (yyyyMMdd)
RULE_IGNOREMISSINGGROUP	CHEE	S	0	0 - no (reject) 1 - yes (warning, no rule)
GROUP_CREATE	CHE	S		1 - create RULE_GROUP entry 0 - not
GROUP_NAME.*	CHE	S		RULE_GROUP Name
GROUP_NAME.0	CHE	S	1	number of RULE_GROUPs to create
GROUP_NAME	CHE	S		default for GROUP_NAME.*
GROUPIN_CREATE	CHE	S	0	1 - create GROUPIN entry 0 - not
GROUPIN_DESC	CHE	S		GROUPIN Description
GROUPIN_NAME	CHE	S		GROUPIN Name
GROUPINEXTRAWHERE	CHE	S	empty	Additional where clause for deletion of complex rules (table GROUPIN)
Signatory Settings				
BIRTHDATE	CHME	S	empty	Birthdate (yyyyMMdd)
INSTRCODE	CHME	S	00	Instruction code
INSTRDATETO	CHME	S	empty	Instruction valid till (yyyyMMdd)
INSTRTEXT	CHME	S	empty	Instruction text
INSTRUID	CHME	S	empty	Instruction userid
MNAME	CME	S	empty	Signatory middle name
LATIN	CME	S	1	Signatory language type

Key	Action	Target	Default	Description
MONOCLEAN	CHME	SV	0	0 - no cleaning 1 - cleaning
PARMCLEAN	CHME	SV	8007D06	Gray image cleaning parameters
PARMCLIP	CHSE	SV	empty	Clipping parameters
QUALITY	CHM	SV	1	Signature quailty
SIGNATORY_ROLE	CHME	S	01	Signatory Role
SIGTYPE	CHME	S	S	Signatory type
DATESIGNED	CHME	S	empty	Date signed (yyyyMMdd)
POSITION	CHME	S	empty	Position
COUNTER_USED	CHME	S	empty	How often used
DATESCANNED	CHME	S	empty	Date scanned (yyyyMMdd)
DATESIGNED	CHME	S	empty	Date signed (yyyyMMdd)
VARVALIDFROM	CHME	S	empty	Variant valid from (yyyy-MM-dd)
UNLINKVARIANTS	CHE	S	0	0 - no 1 - yes (only for multiupdate and a new signature)
UNIQUEPID	C	S	0	Unique personal Id: 0 no 1 yes (if yes,check if signatory already exists)
CSL.ESCALATION	CH	S	1	Licence check escalation: 1 - Warning 2 - Warning, no writing of the signature 3 - Reject 4 - Error Stop
UPDATEALLSIGNATORIES	CHE	S	0	0 - no 1 - yes
Account Settings				
BRANCHCODE	CMHE	A		Branch code
ASTATUS	CMHE	A		Account status: 0 open, 1 closed
FREETEXT	CMHE	A		Account Freetext
OPENDATE	AMHE	A		Account creation date (yyyyMMdd)
DEMOGRAPHICLOCN	AMHE	A		Demographic location
Other Settings				
ACCOUNTMODEL				0 - Customer model 1 - Account model

Key	Action	Target	Default	Description
ASV	CHME	CAS		ASV Flag
DIFF	HE	CAS	0	0 - update only if there is a difference to the database 1 - update always
CONTROL.ACTION	all	all	S	S - Select N - no database action C - continue (do nothing)
NEWCUSTOMER	XHE	C	empty	If not empty, then copy Customer and delete the old one (for actions H,E)
COPYCUSTOMER	XHE	C	empty	If true, then copy Customer and NO delete of the old one (for actions H,E)
NEWBNO	XHE	C	empty	New BNO for the function "copy Customer and delete the old one (for actions H,E)"
NEWCOUNTRYID	XHE	C	empty	New COUNTRYID for the function "copy Customer and delete the old one (for actions H,E)"
NEWBANKCODE	XHE	C	empty	New BANKCODE for the function "copy Customer and delete the old one (for actions H,E)"
OPID	all	all	ACCOUNTLOADER	Operator name
OPID_VERIFY	all	C	OPID	Verify operator name
TIMESTAMPVERIFY	all	C	TIMESTAMP	Value for the TIME_STAMP_VERIFY column
PROTOCOL	all	all	1	1 - write protocol 0 - don't write
RESTCODE	CME	CA	0	Account and Customer Restriction code
RESTTEXT	CHME	CA		Restriction text
RESTDATETO	CHME	CA		Restriction valid till (yyyyMMdd)
TIMESTAMP	all	all		Current timestamp
TIMESTAMPENTRY	all	all	TIMESTAMP	Value for all TIME_STAMP_ENTRY columns
UPDATEACCOUNT	C	A	empty	1 - perform an update on the existing account 0 - reject the record
VALIDFROM			TIMESTAMP	The valid from date
VALIDTO			empty	If not empty, the VALIDTO date

Key	Action	Target	Default	Description
VERIFY	all	all	empty	If 1, customer must be verified after this change
DB.INSERT				Count of database INSERTs during processing, for SignBase statistics
DB.UPDATE				Count of database UPDATEs during processing, for SignBase statistics
DB.DELETE				Count of database DELETEs during processing, for SignBase statistics

Report file format

All fields are delimited by ";" or the character defined with key reportDelimiter. The format can change project-specific.

Field	Description
BNO	Bank number
Customer	Customer number
Action	I - Inserted U - Updated D - Deleted

ImageLoader

This service program is based on the same code as the AccountLoader, but has a more specialized configuration. The ImageLoader loads checkstock images into the SignBase database. The following features are possible and configurable:

- Comparing a new image with the reference images in the database. Too similar new images are ignored
- Checking amount ranges
- Checking Correction Items
- Checking for unusual sizes
- PAD check
- Pre-PAD check (for performance reasons)
- ASV check
- Serial Number check
- Creating input data for a SignatureReferenceFilter
- Variants check
- Historizing of old entries when the maximum count of checkstock images is reached

After successfully processing a data file, this data file and it's activate file, so far specified, will be deleted or renamed. If the processing of a data file failed, the data file will be renamed to avoid a second processing and the log will be written to a file with the same name as the data file, but with a different extension.

Configuration of the ImageLoader

Additional to the above described configuration of all main configuration files there are the following keys:

Key	Default	Description
commitLine	true	true – every input record is committed to the database, otherwise not (but maybe via function COMMIT)
action(f)	C	The action to be performed. This can be only: C - Create data
trailerRecord	false	If the data file has a trailing record, which is no regular data record, the contents of the last line is stored in the hashtable in key TRAILER.
activateSuffix(f)	value of dataSuffix	The suffix of an activate file with the same name as the data file. The existence of this file allows the processing of the data file.
real.RenameSuffix		The renameSuffix for the case that a file could be processed properly, otherwise the renameSuffix could be an other suffix defined in key renameSuffix.
REPORT.ReportPath		Path of the report files

PADEngine		Defines the engine to be used for PAD check. The name of this key is part of the key defining the class to be used:
		`<name>PAD = <classname>`
		The following engines are available:
		`GIAPAD = de.softpro.signplus.service.GIAPAD`
		`nonePAD = de.softpro.signplus.service.FunctionDummy (this engine does nothing)`

referenceEvaluator	CheckHitrate	Name of the class that performs the evaluation of the hit rate, must be derived from CheckHitrate.
PADkey.0	0	Count of keys containing keywords for PAD
PADkey.1		1st keyword for PAD
PADkey.2		2nd keyword for PAD etc.
protocolInsert	false	If true, then the count of inserts is written to the log when the program stops.
protocolUpdate	false	If true, then the count of updates is written to the log when the program stops.
protocolDelete	false	If true, then the count of deletes is written to the log when the program stops.

Key	Default	Description
changeUser	0	1 - update the CUSTOMER_CHGUSER table after changing a customer object 0 - don’t update
protocol	0	1 - update the PROTOCOL table after a change 0 - don’t update
SRFstoreIRD	1	1 - store variants from an IRD 0 - don’t store
customerlockTableResource.0	0	Count of table resource files for the customer locking mechanism
customerlockTableResource.1		1st table resource file for the customer locking mechanism
customerlockTableResource.2		2nd table resource file for the customer locking mechanism etc.
SBStatisticsTableResource		Table resource file to perform the SignBase statistics
check.AmountRange		1 - amount range check 0 - no amount range check
ignoreRange.0		Count of amount ignore ranges
ignoreRange.1		1st ignore range. Format: “<minAmount>,<maxAmount>”, all amounts in cent.
ignoreRangePrivate.0		Count of amount ignore ranges for private accounts
ignoreRangePrivate.1		1st ignore range for private accounts
ignoreRangeCorporate.0		Count of amount ignore ranges for corporate accounts
ignoreRangeCorporate.1		1st ignore range for corporate accounts
ignoreRangeCorporate.2		2nd ignore range for corporate accounts
ignoreRangeOther.0	0	Count of amount ignore ranges for other accounts
ignoreRangeOther.1		1st ignore range for other accounts
ignoreRangeOther.2		2nd ignore range for other accounts
check.IRDCheck		1 - IRD (Image Replacement Document) check 0 - no IRD check
check.CorrectionItemCheck		1 - Correction Item check 0 - no Correction Item check
check.UnusualSizeCheck		1 - Unusual Size check 0 - no Unusual Size check
check.PADCheck		1 - PAD (Pre Authorized Draft) check 0 - no PAD check
check.prePADCheck		1 - Pre PAD check 0 - no Pre PAD check (only for performance reasons)
check.PADCleanedSizeMin		Minimum size (in bytes) of a cleaned PAD

Key	Default	Description
check.PADCleanedSizeMax		Maximum size (in bytes) of a cleaned PAD
check.ASVCheck		1 - ASV check (ACCOUNT.ASV=1 (ASV=0: ignore)) 0 - no ASV check
check.SerialNoCheck		1 - Serial Number check 0 - no Serial Number check
check.minSerialNo		Minimum Serial Number
check.EngineCheck		0 - don’t compare
check.VariantsCheck		1 - check if the maximum count of variants is reached 0 - don’t check
minAgeOldVariants		Minimum age of a variant to be allowed to be deleted
check.CreateSRFData		1 - create SRF data 0 - don’t create SRF data

Configuration of the ImageLoader in service.properties

Beside the settings of the main configuration file the following settings are possible:

Key	Default	Description
IL.PADLevel	0	Confidence level for PAD
IL.deltaSize	20	Maximum difference of width and height in pixels for comparing 2 images
IL.FPLevel	0	Confidence level for checkstock compare
IL.maxImages	0	Maximum count of checkstock images
maxImagesPrivate	IL.maxImages	Maximum count of checkstock images for private accounts
IL.maxImagesCorporate	IL.maxImages	Maximum count of checkstock images for corporate accounts
IL.maxImagesOther	IL.maxImages	Maximum count of checkstock images for other accounts
IL.minAgeOldImages	0	Minimum age of old images in days
IL.deleteOldImages	0	1 - delete old images if the maximum is reached 0 - don’t delete old images
IL.maxVariants	0	Maximum Variants per customer and signatory
IL.maxVariantsPrivate	IL.maxVariants	Maximum Variants per customer and signatory for private accounts

Key	Default	Description
IL.maxVariantsCorporate	IL.maxVariants	Maximum Variants per customer and signatory for corporate accounts
IL.maxVariantsOther	IL.maxVariants	Maximum Variants per customer and signatory for other accounts
IL.SRFTempSuffix	.sr0	Temporary data suffix for SRF files
IL.SRFDataSuffix		Final data suffix for SRF files
IL.minResolution	200	Minimum resolution for processing an image
IL.keepFraudulentImages	false	true - images marked as fraudulent are completely ignored
IL.validFrom	0	Count of days starting from today when a new checkstock image becomes valid
IL.minCountValidImages	1	Old image are deleted only when the count of valid images is bigger than this setting
IL.minCountValidImagesPrivate	IL.minCountValidImages	Minimum count of valid images for private accounts
IL.minCountValidImagesCorporate	IL.minCountValidImages	Minimum count of valid images for corporate accounts
IL.minCountValidImagesOther	IL.minCountValidImages	Minimum count of valid images for other accounts
IL.SRFstoreIRD	0	1 - store also IRD's as new variants 0 - ignore IRDs
IL.BNOQuery	false	true - BNO is queried from the database false - BNO is already defined
IL.CustomerQuery	false	true - CUSTOMERNO is queried from the database, this makes sense only for customer model false - CUSTOMERNO is already defined

Special keys in the hashtable

The following keys are created automatically after processing of every table resource file:

Key	Description
LAST.RESULT	Number of rows changed by the last UPDATE, INSERT or DELETE.
LAST.TABLE	Name of the table of the last UPDATE, INSERT or DELETE.
TABLE	Name of the table of the last successful UPDATE, INSERT or DELETE.
REPORT.CHANGED	1 after a successful INSERT, UPDATE or DELETE statement, otherwise 0.

Key	Description
EXTRA.INFO	Additional information appended to the log when the AccountLoader stops with a message like “I 2012-06-19 14:28:08.421 -: stopped after 8 records. elapsed time: 0:02:25.541”

This key is created after opening a data file:

Key	Description
TRAILER	The last record of the data file if the key trailerRecord is true, otherwise TRAILER is empty.

The following keys must be defined after reading a record from the data file and processing ILSetValues.properties:

Key	Description
BNO	
COUNTRYID	
BANKCODE	
CUSTOMERNO	
DOCREFNO	Document reference number
CHECKITEM	1 - check, otherwise don't check
SERIALNO	Serial number
ACCOUNT	
AMOUNT	In cent
CURRENCY	EUR, USD, ...
CLEARDATE	Clearing date
FORMTYPE	0 - normal 3 - correction item 4 - IRD
IMAGEFILE	Name of the image file
IMAGEOFFSET	Offset of the image inside the image file, starting with 1
IMAGELENGTH	Length of the image, if 0 then IMAGEOFFSET is the page number of a multi tiff file
FRONTIMAGE	Front image or empty if not available
TIMESTAMP	Optional (Default: current time)
TEMPLATE	New IMAGE_PARM for the update of a STOCKIMAGE, currently disabled

Report file format

All fields are delimited by “;” or the character defined with key reportDelimiter. The format can change project-specific.

Field	Description
BNO	Bank number
Customer	Customer number
Type	Customer Type
DocID	Unique document ID
NumImg	Number of incumbent reference check images
Action	I - new reference inserted C - no reference inserted - account complete Q - no reference inserted - bad quality R - no reference inserted - item image considered risky P - reference parameter inserted (check image only) N - not inserted S - not selected for insert J - rejected A - amount too little T - image too similar to an existing one B - missing SignBase data V - Account blocked
ImgNo	Reference image number of new inserted image. (not added => " ")
Match	Best match rate against incumbent reference " " - no incumbent reference or no validation performed
MatchImg	Reference number for above match rate
The following fields are delivered only if full report is enabled (reportOnlyDB=false):	
Quality	00 - OK 01 - Unusual size 02 - PAD detected 03 - IRD 04 - serial number too small or account complete 05 - no Image 06 - correction Item 07 - bad resolution
Test	Confidence level from PAD-test for check images " " - no test

DataViewer

The DataViewer is a tool for visual controlling service programs that work with input files, especially those containing images. With e DataViewer it is easier to check the configuration of the service program and also the check the input files.

With the "file-open" dialog a data file can be loaded. All lines of this data file are displayed. In the main configuration file can be configured, which columns to display. By double-clicking a specific line, all column's values of all tables of this line are shown on another table, in the same way as in the actual service program. You can display an image by double-clicking on a table-row that contains an image. In a second window the chosen image will be displayed. If this image contains a signature snippet, the border of this snippet is shown with a colored frame. Changing this frame by mouse-movements results also in a change of the coordinates of this snippet. The default for the settings of the coordinates are in table SC_INTERFACE (F_UPPER_LEFT_X1, F_UPPER_LEFT_Y1, F_LOWER_RIGHT_X1, F_LOWER_RIGHT_Y1).

Configuration of the DataViewer

Additional to the above described configuration of all main configuration files there are the following keys:

Key	Default	Description
listColumns	0	Number of columns for the overview about all rows of a data file minus 1 (the first column is always the line number)
listColumn1		The name of column 2 (the first column has always the name “No.”)
listColumn2		The name of column 3 etc.
listFormula1		The formula for calculating the value for column 2 (the formula for column 1 is always “${LINE.NUMBER}”, i.e. the record number). The formula can contain variables of the form ${name…} where name is the name of a hashtable key. A name can be followed by functions described in the chapter Formulas.
listFormula2		The formula for calculating the value for column 3 and so on

Key	Default	Description
font	Windows-setting	The font for the image display area Syntax „name,style,size“ where: name the name of the font style the style of the font: 0 - normal 1 - bold 2 - italic 3 - bold and italic size the size of the font in pixels Example „Helvetica,0,16“ Example „Helvetica,0,16“
datDescription	„SignCheck Data File (*dataSuffix)“	The description of the file extension dataSuffix (for the filter of the file chooser dialog)
clips	0	Number of signatures on all images
clip1Image	SC_IMAGE.FRONT_IMAGEM	Name of the column of the 1st image. Notation for all columns is: <table>.<column>
clip2Image	SC_IMAGE.FRONT_IMAGEG	Name of the column of the 2nd image etc.
clip1Number	1	Number of the signature snippet on image 1. A maximum of 2 snippets per image is possible.
clip2Number	1	Number of the signature snippet on image 2 etc.
clip1Color		The color definition for the rectangle around the 1st signature snippet. Format is: <red>,<blue>,<green> where every color can go from 0 to 255.
clip2Color		The color definition for the rectangle around the 2nd signature snippet etc.
clip1Left	SC_IMAGE.F_UPPER_LEFT_X1	Name of the column containing the coordinate of the left edge of the snippet of image 1

Key	Default	Description
clip2Left	SC_IMAGE.F_UPPER_LEFT_X1	Name of the column containing the coordinate of the left edge of the snippet of image 2 etc.
clip1Top	SC_IMAGE.F_UPPER_LEFT_Y1	Name of the column containing the coordinate of the upper edge of the snippet of image 1
clip2Top	SC_IMAGE.F_UPPER_LEFT_Y1	Name of the column containing the coordinate of the upper edge of the snippet of image 2 etc.
clip1Right	SC_IMAGE.F_LOWER_RIGHT_X1	Name of the column containing the coordinate of the right edge of the snippet of image 1
clip2Right	SC_IMAGE.F_LOWER_RIGHT_X1	Name of the column containing the coordinate of the right edge of the snippet of image 2 etc.
clip1Bottom	SC_IMAGE.F_LOWER_RIGHT_Y1	Name of the column containing the coordinate of the bottom edge of the snippet of image 1
clip2Bottom	SC_IMAGE.F_LOWER_RIGHT_Y1	Name of the column containing the coordinate of the bottom edge of the snippet of image 2 etc.
clip1ResX	SC_IMAGE.F_NORM_RES_X	Name of the column containing the X-resolution of image 1
clip2ResX	SC_IMAGE.F_NORM_RES_X	Name of the column containing the X-resolution of image 2 etc.
clip1ResY	SC_IMAGE.F_NORM_RES_Y	Name of the column containing the Y-resolution of image 1
clip2ResY	SC_IMAGE.F_NORM_RES_Y	Name of the column containing the Y-resolution of image 2 etc.
resubstitute.0	0	Number of columns that have to be substituted again after a rectangle change on the GUI. This is helpful when configuring the clipping rectangle manually.
resubstitute.<n>		Name of the n-th column that has to be substituted again, n=1 to <resubstitute.0>. Notation for all columns is: `<table>.<column>`
frameColumn.0	0	Number of columns that contain images with frames, e.g. after a MONOCLEAN or a LINESEARCH function
frameColumn.<n>		Name of the n-th column that contains an image with frames, n=1 to <frameColumn.0>. Notation for all columns is: `<table>.<column>`

Key	Default	Description
frameWeight.<n>		Hashtable basename for the n-th column containing the weights of the frames of the image, n=1 to <frameColumn.0>.
frameWeightMax.<n>		Maximum value for the weights of the frames for the n-th column, n=1 to <frameColumn.0>.
frameList.<n>		Hashtable basename for the n-th column containing the frames of the image, n=1 to <frameColumn.0>. Every frame key contains 4 comma-separated values with the position of the frame in pixels, with the resolution of the image. Format is: `<left>,<top>,<right>,<bottom>`
frameColorHigh.<n>		Hashtable basename for the n-th column containing the color definition for the frames of the image, for high weights, n=1 to <frameColumn.0>. Format is: `<red>,<blue>,<green>` where every color can go from 0 to 255. By using the cleaning slider on the GUI it is possible to change the color of the frames, depending on their weights.
frameColorLow.<n>		Hashtable basename for the n-th column containing the color definition for the frames of the image, for low weights, n=1 to <frameColumn.0>.

SignatureReferenceFilter (SRF)

Variants among the reference signatures in the SignBase database can be used to bridge gaps during the long recording phase when introducing automatic signature verification in payment transactions. These can be entered with a variant load (also in batch operation). Before these variants can be used, they must be assigned to existing authorized signatories and those signatories must have information on signing rules. New Variants are compared with the existing references to prevent the loading of any images that are similar to the reference signatures.

“Similar” in this sense is used to mean that the result match rate of two signatures is higher than the configured setting of referenceMatchRate in the property file.

The SRF supports two modes, on-line mode and off-line mode. The fundamental differences between the two modes are, that the SRF in on-line mode works together with the SignBase database. The SRF in off-line mode doesn't work with any databases and doesn’t need any connections to a server.

The SignatureRereferenceFilter reads the records for each account in a vector for comparison.
If the sorting is set, than the signatures can be sorted according to a date e.g. to scan date: referenceDateSequence= SCAN_DATE.

The following DATE fields are supported: NONE|SCAN_DATE|SIGN_DATE|PN_DATE|PROC_DATE|CLEAR_DATE

Default: the records won't be sorted.

The signatures comparison is performed using the SIVAL program.

If SIVAL recognizes two signatures as similar, the following signatures from the input file will be not accepted:

1. The next similar signature after the first (accepted). In Offline mode by default setting the first signature is always accepted.
2. The signature with an oldest date, if the key referenceDateSequence is set.
3. The signatures with a minimum compare result, if SRFmodifiedLogic is set.

The similar signatures will be ignored and not loaded in database. The not similar signatures will be saved into the SignBase database as variants in on-line mode, or written into the resulting file in off-line mode.

The resulting filtered file in off-line mode must be used as an input file for another SRF to insert the images into the database.

For every not accepted signatures a log with trace level SELECTION will be created:

Variant in line 1 matched with variant in line 0 with matchrate AA and will be ignored for account 00042404

For each accepted signature image written to the new result file (in off-line mode) or to the SignBase database (in on-line mode), the program will write a report line to its log (if the key sivalMatchRate is not set).

"Accepted: 'CustomerNo', 'AccountNo',

"Accepted: 'CustomerNo', 'AccountNo', m1

"Accepted: 'CustomerNo', 'AccountNo', m1,m2,

…

m1,m2,m3… are the match rates.

Note Max. 2 SignatureReferenceFilter are allowed to be run on one PC at the same time, because the SignatureReferenceFilter is actually a special SignBase client and needs therefore the SignBase server.

On-line mode

SRF in on-line mode works with the SignBase database, therefore it needs a connection to the server and database.

The on-line mode supports the account and customer model. In the customer model the customer number will be gotten from the database for every new account number. A maximum of three signatures can be

compared for each record (the first and the second signature on the front page and the third signature on the back page of check).

All signatures from an input file will be loaded per account to a vector, additionally the signatures from the SignBase database for this customer will be loaded to this vector. All signatures will be compared to each other and all not similar will be loaded to the database.

For a better performance all records in the input file should be sorted to the account. In this case all signatures will be loaded line-by-line until SRF find a record with a new account. After all records for an account are loaded, a customer for this account group will be get from SignBase database and all signatures from database will be loaded for comparison. If the input records are not sorted the customer search will be happened for each new account and that will decrease the performance.

Off-line mode

Because die off-line mode doesn’t support the connection to the database, to get a customer number from, the SignatureReferenceFilter works only in account model. Also, only one signature can be processed for each record from the input file (first or second on the front page of check, or one on the backside of check).

The reason for this is that the records (if accepted) should be written in a result file. For each accepted record from the input ‘dat-file’ one record is inserted in the output result file.

The compare algorithm is almost the same as with the on-line mode. The difference is that the signature will not be inserted into the database, but will be inserted (if accepted) in the result file.

All records with the accepted signatures will be inserted into the output result file. This reduced result file can be used later by a SRF in online modus.

In off-line mode the "first" signature image is always accepted and is not subject to any comparisons and so will have no match rate numbers in its log record (by Barclays the signatures per customer are sorted to date, the first is the oldest). But if you use the modify logic SRFmodifiedLogic, the first accepted signature will be with a best match rate result.

All records in the input file should be sorted to the account. SRF reads all records line-by-line from the input file for each account group and compares between themselves. If the records are not sorted to an account, the comparison for each account group can not take place and the similar signatures won’t be rejected.

Clipping

The SIVAL checks the signature at the left edge very strict. If a signature at the left edge has a spot, the SIVAL can’t recognize it as a spot to clean this, but recognizes it as a part of the signature. Because of this the signature won’t be cleaned very well at the left edge and the result of the match rate could be not enough to recognize the similarity.

In order to avoid this the function clipping was created. This means that this function can’t really improve this situation, but sometimes makes worse, because the important initials at the left edge could be deleted. So, it should be really verified whether the clipping function can be used or not.

The task of clipping function is, that two signatures will be cut and checked four times and only the best match rate result will be taken over. The size of signature or image by writing into the database won't be changed.

If the comparing result during the four times process gets a match rate of 100% (this is a best match rate result) the comparing process stops with this pair of signatures and begins with the next one.

Example 1

```
compare signature 0 (100%) with 1 (100%), match rate is 79
compare signature 0 (100%) with 1 (90%), match rate is 94
compare signature 0 (90%) with 1 (100%), match rate is 100
compared 0 signature to 1, match rate 100
```

Example 2

```
Compare signature 0 (100%) with 3 (100%), match rate is 88
Compare signature 0 (100%) with 3 (90%), match rate is 12
Compare signature 0 (90%) with 3 (100%), match rate is 40
Compare signature 0 (90%) with 3 (90%), match rate is 85
Compared 0 signature to 3, match rate 88
```

Function modified logic

In deference to the default setting here all signatures of the customer/account will be written in one vector and than die match rates and the output logs as well will be evaluated. For every signature a value "*max nodes*" will be created, that shows how often the signature with others signatures matches.

The other value is a *max match rate sum* per signature (match rate between the current signature and other signatures). The signature with a *max match rate sum* or *max nodes* will be accepted as the first one. In this case the value "1" will be set for the accepted signature. Then the accepted signature will be compared with the other signatures to similarity.

We compare four signatures. The signature 0 is accepted as the first one (s. bellow). This signature will be compared with the signatures 1, 2 and 3. Where the accepted signature matches (that mean that the signature are similar) a value "-1" will be set.

The signature 0 matches with the signature 2 (100%) and with the signature 3 (88%), so the signature 2 and 3 will be not accepted (they are similar to the signature 0).

In the table below you see, that only the signature 0 and 1 are accepted, because they don't match. For this signatures, which result match rate is smaller than match rate limit the *match rate of the indirect similarity* will be evaluated.

signatureNo	0	1	2	3	n	Max mr
0	**1**	79/83	100	88	2	267
1	79/83	**X**	83	21	1	183
2	100	83	**-1**	0	2	183
3	88	21	0	**-1**	1	109

result: *accepted variant 0, "AccountNo", n=2, mr=267*

signatureNo	0	1	2	3	n	Max mr
0	**1**	79/83	100	88	2	267
1	79/83	**X**	83	21/79	1	183
2	100	83	**-1**	0	2	183
3	88	21/79	0	**-1**	1	109

result: *accepted variant 1, “AccountNo”, n=1, mr=183*

n- number, how oft the signature matches

mr- match rate

Match rate of the indirect similarity

The meaning of the match rate of the indirect similarity is, that the not matched signatures could have the similarity through the other signatures.

So, e.g. the signature 0 doesn’t match with the signature 1, because it has a match rate of 79% (79%<80%), but we know that the signature 1 matches with the signature 2 with a match rate of 83% and the signature 2 matches with the signature 0 with a match rate of 100%. From this follows that the signatures 1 and 2 belongs to the same person, like the signature 2 and 0, too. The match rate of the indirect similarity will shown by the slash in the log.

Accepted:, „KontoNo“,

Accepted:, „KontoNo“, 79/83

The meaning here is that the signature 0 and 1 has a similarity of 79%, but through the indirect similarity the match rate is 83%. In this case the operator could decide to accept or refuse the signature 1.

Configuration of the SignatureReferenceFilter

Additional to the above described configuration of all main configuration files there are the following keys.

Keys, that are marked with (f) can be defined with a formula:

```
${[<name>]<delm>[<function1>][<delm><function2>…]… }
```

where:

`<name>`

is a key from a hashtable

`<delm>`

is one of the characters "|", "?" or ":"

`<functionx>`

is one of the defined functions, analog to the formulas in the table properties files

See chapter Formulas.

Example

```
maxVariants=${BNO|TEST$*<=305?FMT3:FMT2}:
```

Key	Default	Description
user		The SignBase user id
password		The SignBase user id's password. If the user/password combination is not valid, a logon dialog is shown to let the user retry a logon.
operator(f)	user	If defined, this name will be the userid for all database actions.
autoSetSignature	false	If set to true AND there is only one signatory for the current customer AND this signatory has no signature yet or autoHistorizeSignature is also set to true, then a signature is created instead of a variant.
autoHistorizeSignature	false	If set to true AND there is only one signatory for the current customer AND autoSetSignature is also set to true, then the possibly existing old signature and all variants are historized.
checkBNOLicence	true	If set to true, then a license check is performed in dependency of the BNO. If the license check fails, this record will be rejected.
useRegExpForDataSuffix	not defined	If defined, data files are not filtered by a suffix, but by matching the regular expression in dataSuffix. Example `dataSuffix = [Dd]\\d\\d\\d\\.\\d\\d\\d\\d$` useRegExpForDataSuffix=true
substituteAlways	false	If set to false, then only the table containing the selection stuff is substituted. After a positive checking of these criteria the remaining tables are substituted. This happens only for performance reasons. If set to true, all tables are substituted immediately.
createDummySignatory(f)	false	If set to true, a so-called dummy signatory is created for the current customer, so far this dummy did not exist.

Key	Default	Description
assignToDummySignatory(f)	false	If set to true, a so-called dummy signatory is created for the current customer, so far this dummy did not exist AND each variant will be assigned to this dummy.
assignToPersonalId	false	true - bind new variants to a specific Personal Id false - no bind
deleteOldVariants	false	If set to true, the oldest variant will be deleted if the maximum number of variants for a customer is reached, before adding the new variant.
minAgeOldVariants	0	If deleteOldVariants is set to true, the oldest variant will be deleted only if the maximum number of variants for a customer is reached and the age of the oldest variant in days has at least the value of minAgeOldVariants.
searchForClosedAccounts	false	true - search also for closed accounts false - search only for open accounts
createCustomer	false	If set to true, a customer will be created for the current variant, if this customer does not exist.
createAccount	false	If set to true, an account will be created for the current variant, if this account does not exist.
createStockImage	false	If set to true, the front images are stored as checkstock references.
reconnectWait	10	The number of seconds to wait before trying to reconnect to the server, if this connection was lost.
referenceEvaluator	CheckHitrate	Name of the class that performs the evaluation of the hit rate, must be derived from CheckHitrate.
maxVariants(f)	0	The maximum number of variants for a customer. This is an average value dependent on the number of signatories of a customer. If this number is set to 3 and a customer has 4 signatories, then this limit is reached, when there are 12 variants, regardless to whom assigned.
maxVariantsPrivate(f)	maxVariants	The maximum number of variants for a private customer.
maxVariantsCorporate(f)	maxVariants	The maximum number of variants for a corporate customer.

Key	Default	Description
maxVariantsOther(f)	maxVariants	The maximum number of variants for neither private nor corporate customers.
maxVariantsAutoAssign(f)	maxVariants	The maximum number of variants for a customer if autoassign=true.
maxVariantsAutoAssignPrivate(f)	maxVariants	The maximum number of variants for a private customer if autoassign=true.
maxVariantsAutoAssignCorporate(f)	maxVariants	The maximum number of variants for a corporate customer if autoassign=true.
maxVariantsAutoAssignOther(f)	maxVariants	The maximum number of variants neither private nor corporate customers if autoassign=true.
autoAssign	false	If set to true and the current customer has only one signatory, the variant will automatically assigned to this signatory.
autoAssignPrivate	autoAssign	If set to true and the current customer has only one signatory and is private, the variant will automatically assigned to this signatory.
autoAssignCorporate	autoAssign	If set to true and the current customer has only one signatory and is a corporate customer, the variant will automatically assigned to this signatory.
autoAssignOther	autoAssign	If set to true and the current customer has only one signatory and is neither private nor corporate, the variant will automatically assigned to this signatory.
assignToOldestSignatory	false	If set to true, every variant will be assigned to the oldest signatory of this customer. The oldest signatory is the one with the oldest TIME-STAMP_ENTRY.
minAmount_<currency>	all records are processed	The minimum amount in the currency <currency>. Is a record’s amount less than this amount, it will be ignored.
minAmount_<currency>_BNO_<bno>	minAmount_<currency>	The minimum amount in the currency <currency> for BNO <bno>. Is a record’s amount less than this amount and the BNO is <bno>, it will be ignored.

Key	Default	Description
minAmount_<currency>_BNO_<bno> _BANKCODE_<bankcode>	minAmount_<currency> _BNO_<bno>	The minimum amount in the currency <currency> for BNO <bno> and for BANKCODE <bankcode>. Is a record's amount less than this amount and the BNO is <bno> and the for BANKCODE is <bankcode>, it will be ignored.
referenceOnlineMode	true	Switching between the two modes, on-line and off-line, see the description for the modes below.
referenceMatchRate	80	The match rate limit. If two signatures have a match rate of 12% (12%<80%), these signatures will be recognized by the SIVAL program as not similar (like from different persons). In this case both signatures will be accepted. But if two signatures have a match rate of 90% (90>80), the SIVAL program recognizes these signatures as similar (from the same person) and the signature with the oldest date will be deleted.
sivalMatchRate	NONE	Analogical to the SignCheck automats the Match Rate will be appeared in a log file as a big letters instead of numbers from 0 to 100% (e.g. 100=AA, 90=A1, 50=B4, .. see the file „asv_rate.txt"). This key has a higher priority against the key referenceMatchRate.
referenceDateSequence	NONE	The signatures can be sorted to ensure that the "newest" signature image per Customer/Account contained in the input file will be selected in the event that more than one similar image is present in the input file. The SRF supports any of the following date fields: NONE|SCAN_DATE|SIGN_DATE| PN_DATE| PROC_DATE|CLEAR_DATE| SCAN_DATE| NONE[no sorting]
offlineResultExtension	.flt	Only for off-line mode: Directory for the result file.
offlineResultDirectory	Path from key "dataDir"	Only for off-line mode: Path for the result file.

Key	Default	Description
SRFClipping	no	Two signatures will be compared 4 times instead of one time, this will get a best result for the Match Rate, see SRFClippingDepth.
SRFClippingDepth	10	Only if SRFClipping=true How many percent will be cut from the signature, e.g. if SRFClippingDepth=10 100%Reference to 100%Signature 100%Reference to 90%Signature 90%Reference to 100%Signature 90%Reference to 90%Signature
SRFmodifiedLogic	false	Accept not the first signature but the signature with a most match rates correct the match rates through the indirect similarity function.
maxCompare	10	The maximum number of signatures from a data file that are compared against each other and those in the database. If the actual number of signatures of a customer in the data file is bigger, the signatures are processed in blocks.
ignoreDualSigners	false	If set to true and a customer has at least 1 complex rule like a group-rule or a collective rule then no new variants are added to this customer.

Signatory extensions

If the data model of SignBase has extensions for signatories, then these additional fields have also to be filled for variants. These additional fields are defined in the 2nd table properties file, as additional (pseudo-) columns. The number of extension fields is defined in the column with the name SIGNATORY_EXTENSIONS. Depending on this number there are the columns SIGNATORY_EXTENSION1, SIGNATORY_EXTENSION2 etc. If the signatory has no extensions, SIGNATORY_EXTENSIONS should be set to 0 or should not be defined at all. The value for each signatory extension is as follows:

```
< FIELDID>,< CONTENTTYPE>,< CONTENT>
```

FIELDID	Field name - unique in Base Object Type: mapping from Id to bank specific meaning in client

CONTENTTYPE	Data type for content: '1'=Char '2'=Char-Binary '3'=SmallInt '4'=Integer '5'=Decimal '6'=Date '7'=String ...
CONTENT	Field content: always stored as CHARACTER

Precedence of assignment of variants

The assignment of variants to signatories is controlled by the keys *autoassignXXX* (XXX = 'Private', 'Corporate', 'Other' or <empty>), *assignToOldestSignatory* and *assignToDummySignatory*. Every variant is assigned to only one signatory. The following list determines the precedence of the keys above:

1. *autoassignXXX*
2. *assignToOldestSignatory*
3. *assignToDummySignatory*

This means, that an assignment to the so-called "dummy signatory" takes place only when neither *autoassignXXX* nor assignToOldestSignatory was possible or configured resp.

Special columns in the table properties files

The following columns should be defined in the 1st table properties file:

Column	Description
IMAGEFILE	Name of the image file
IMAGEOFFSET	Offset of the image inside the image file, starting with 1
IMAGELENGTH	Length of the image, if 0 then IMAGEOFFSET is the page number of a multi tiff file
<t>SIG<n>	Contains a signature image, <t>=G or M (gray or mono), <n>=1,2,3
<t>WIDTH<n>	Contains the width of the signature image <t>SIG<n>, <t>=G or M (gray or mono), <n>=1,2,3
<t>HEIGHT<n>	Contains the height of the signature image <t>SIG<n>, <t>=G or M (gray or mono), <n>=1,2,3
<t>RESX<n>	Contains the X-resolution of the signature image <t>SIG<n>, <t>=G or M (gray or mono), <n>=1,2,3
<t>RESY<n>	Contains the Y-resolution of the signature image <t>SIG<n>, <t>=G or M (gray or mono), <n>=1,2,3

Contains the Y-resolution of the signature image <t>SIG<n>, <t>=G or M (gray or mono), <n>=1,2,3

Column	Description
BNO	
COUNTRYID	
BANKCODE	
CUSTOMERNO	
DOCREFNO	Document reference number
CHECKITEM	1 - check, otherwise don't check
VAR_BATCH_LOAD	1 - this item is intended for creating of variants, otherwise ignore it
ACCOUNT	
AMOUNT_LOCAL	In cent
CURRENCY_LOCAL	EUR, USD, ...
SCAN_DATE	Scanning date
SIGN_DATE	Signing date
FORM_TYPE	0 - normal, 3 - correction item, 4 - IRD, P - the document is ignored
VARVALIDFROM	Count of days starting from today when the variant becomes valid
SIGNATORY_EXTENSIONS	Count of signatory extensions
SIGNATORY_EXTENSION1	Value of the 1st signatory extension in the format "id,type,value"
SIGNATORY_EXTENSION2	Value of the 2nd signatory extension in the format "id,type,value" etc.

Special keys in the hashtable

The following keys are created automatically after processing of every table resource file:

Key	Description
CUSTOMERTYPE	The customer’s customer type, 1 character

Report file format

All fields are delimited by ";" or the character defined with key reportDelimiter. The format can change project-specific.

Field	Description
BNO	Bank number
Customer	Customer number
Type	Customer Type
DocID	Unique document ID
NumSugn	Number of incumbent reference signatures

Field	Description
Action	I - new reference inserted O - new reference inserted AND old reference removed C - no reference inserted - account complete Q - no reference inserted - bad quality R - no reference inserted - item image considered risky N - not inserted S - not selected for insert J - rejected A - amount too little T - signature too similar to an existing one B - missing SignBase data D - item not inserted because it is an IRD U - no reference inserted - not assigned to a Signatory
SignNo	Reference signature number of new inserted signature. (not added => “ “)
Match	Best match rate against incumbent reference signature “ “ - no incumbent reference signature or no validation performed
MatchSign	Reference signature number for above match rate
The following fields are delivered only if full report is enabled (reportOnlyDB=false)	
Quality	00 - OK 01 - No image left after clipping white space 02 - Snippet height to low 03 - Snippet width to slim 04 - Too few pixels left in image 05 - Not enough parameters found in snippet 06 - Too much pixels left in image
Test	Snippet size from signature existence test for signatures “ “ - no test

FraudFeedbackFileLoader

This service program is based on the same code as the AccountLoader, but has a more specialized configuration. The FraudFeedbackFileLoader deletes, blocks or unblocks variants/stockimage references or whole accounts that are known as fraudulent (or no more fraudulent).

After successfully processing a data file, this data file and it’s activate file, so far specified, will be deleted or renamed. If the processing of a data file failed, the data file will be renamed to avoid a second processing and the log will be written to a file with the same name as the data file, but with a different extension.

Configuration of the FraudFeedbackFileLoader

Additional to the above described configuration of all main configuration files there are the following keys:

Key	Default	Description
commitLine	true	true - every input record is committed to the database, otherwise not (but maybe via function COMMIT)
action(f)	C	The action to be performed. This can be only: C - Create data
trailerRecord	false	If the data file has a trailing record, which is no regular data record, the contents of the last line is stored in the hashtable in key TRAILER.
activateSuffix(f)	value of dataSuffix	The suffix of an activate file with the same name as the data file. The existence of this file allows the processing of the data file.
customerlockTableResource.0	0	Count of table resource files for the customer locking mechanism
customerlockTableResource.1		1st table resource file for the customer locking mechanism
customerlockTableResource.2		2nd table resource file for the customer locking mechanism etc.
SBStatisticsTableResource		Table resource file to perform the SignBase statistics
protocolInsert	false	If true, then the count of inserts is written to the log when the program stops.
protocolUpdate	false	If true, then the count of updates is written to the log when the program stops.
protocolDelete	false	If true, then the count of deletes is written to the log when the program stops.
datafile.header	0	If set to 1, then every data file has a header with the format: HYYYY/MM/DD, e.g. H2015/10/22
datafile.trailer	0	If set to 1, then every data file has a trailer with the format: TNN,NNN,NNN (commas and leading zeroes), e.g. T00,000,001 (the number of data records)

Special keys in the hashtable

The following keys are created automatically after processing of every table resource file:

Key	Description
LAST.RESULT	Number of rows changed by the last UPDATE, INSERT or DELETE.
LAST.TABLE	Name of the table of the last UPDATE, INSERT or DELETE.
TABLE	Name of the table of the last successful UPDATE, INSERT or DELETE.
REPORT.CHANGED	1 after a successful INSERT, UPDATE or DELETE statement, otherwise 0.
EXTRA.INFO	Additional information appended to the log when the AccountLoader stops with a message like “I 2012-06-19 14:28:08.421 -: stopped after 8 records. elapsed time: 0:02:25.541”

This key is created after opening a data file:

Key	Description
TRAILER	The last record of the data file if the key trailerRecord is true, otherwise TRAILER is empty.

The following keys can be defined after reading a record from the data file and processing the first table properties file. They control the further processing:

Key	Description
F3ACTION	D - Delete B - Block U - Unblock
ACTION	D - Delete M - Modify
REFERENCETYPE	S - Signature I - Image R - all References A - Account
CONTROL.ACTION	N - no database action S - Select E - Error in the data file
CONTROL.SKIP	F3BA - block account F3BI - block image F3BS - block signatory F3DA - delete account F3DI - delete image F3DS - delete signatory F3End - do nothing

Key	Description
BNO	
COUNTRYID	
BANKCODE	
CUSTOMERNO	
DOCREFNO	Document reference number. It is expected that variants contain the document reference number in the column LNAME and stockimage references in the column DOC_ID.
ACCOUNT	SignBase account
LASTNAME	Lastname for the protocol table
VERIFY_PENDING	The VERIFY_PENDING column of the CUSTOMER
TIMESTAMP	Current timestamp (for DataViewer)
CHANGED	0 - no change in the database 1 - change, has to be set to 0
OPID	Operator ID
CLEARDATE	The date to delete references from a date range to delete references from a date range
DATERANGE	The number of days before and after the CLEARDATE
PROTOCOL	0 - no protocol 1 - write to protocol table
REGULARFILE	0 - no (the data file will not be deleted) 1 - yes
REPORT.PATH	Directory for the report files
REPORT.PN	Process number for the report file name

Report file format

This service program is based on the same code as the AccountLoader, but has a more specialized configuration. The FraudFeedbackFileLoader deletes, blocks or unblocks variants/stockimage references or whole accounts that are known as fraudulent (or no more fraudulent).

After successfully processing a data file, this data file and it's activate file, so far specified, will be deleted or renamed. If the processing of a data file failed, the data file will be renamed to avoid a second processing and the log will be written to a file with the same name as the data file, but with a different extension.

Field	Description
BNO	Bank number
Customer	Customer number
DocID	Unique document ID

Field	Description
Reference-Type	S - Signature I - Image A - Account
Action	D - Logically Deleted B - Blocked from usage R - Account restricted

XML-Loader

The XML-Loader is a program based on a Java SignBase client that is intended for loading XML files into the SignBase database. This happens in batch mode, like AccountLoader or ImageLoader. Input files are XML files according the DTD of signplus.dtd.

Configuration of the XML-Loader

Additional to the above described configuration of all main configuration files there are the following keys:

Key	Default	Description
user	softpro	The user name for logon to the SignBase server
password		The password for logon to the SignBase server
XMLFilter	empty	Name of a class that filters every XML file before loading to SignBase. This class should extend de.softpro.signplus.service.XMLFilter. In this case it is only necessary to write a method "filter". Only if an XMLFilter is specified the signplus.dtd must be present, in the classpath.
fileSort	0	Determines in which order the files in the dataDir are processed. This is important especially in the case that the XML files must be loaded in a determined order, e.g. data can only be added to a customer, when this customer exists. The following settings are possible: 0 - no sort 1 - alphabetical sort 2 - sort per date (FIFO) 3 - random sort <classname> - name of a class that implements the interface SPFileComparator
retrySuffix	empty	Regular expression denoting the file suffix for files that have been previously renamed to the errorSuffix because they could not be processed. After a period of time it is attempted to process these files again.

Key	Default	Description
retryExtension(f)	empty	File name for renaming files that have been renamed to the errorSuffix back to a file name that can be processed according to the dataSuffix.
retryIntervalMillis(f)	0	Date in milliseconds of the last try to process an XML file that has been renamed to the errorSuffix. It is recommended to configure the errorSuffix that way that it contains this date.
retryTimeoutMillis(f)	empty	Date in milliseconds of the first try to process an XML file that has been renamed to the errorSuffix. It is recommended to configure the errorSuffix that way that it contains this date.
retryInterval	1h	Time in milliseconds to specify the retry interval. The following time units are possible: s - second m - minute h - hour d - day w - week
retryTimeout	1w	Time in milliseconds to specify the timeout. An XML file that has been renamed to the errorSuffix before this time will be deleted. The following time units are possible: s - second m - minute h - hour d - day w - week

Additional to the above described configuration of the menu bar there are the following options:

Key	Default	Description
searchSubdirs	off	Search for data files also in subdirectories of dataDir.
simulate	off	Simulate processing of XML files: do not write to the SignBase database

The following keys are created automatically after processing of every XML file:

Key	Description
REPORT.ACTION	P - loading if the XML file succeeded N - loading if the XML file failed U - the XML file is unprocessed D - the XML file could not be processed and has been deleted after the configured timeout R - the XML file has been renamed for a re-processing

Key	Description
REPORT.RESULT	A string containing the result of loading the XML file. If empty, the loading succeeded
REPORT.REST	The number of the file starting with 1 that is tried to re-process again.

Report file format

All fields are delimited by ";" or the character defined with key reportDelimiter. The format can change project-specific.

Field	Description
Filename	Name of the XML file
Action	D - Logically Deleted B - Blocked from usage R - Account restricted
Result	A string containing the result of loading the XML file. If empty, the loading succeeded.

Getter

The SC database is filled with document data from payment transactions. This can take place over a certain time period; event-oriented through entries on the data carrier at the file interface; or by direct read-in of an external medium (e.g. CD-Load).

The access to the database takes place in 2 different ways:

- Via Workflow Server to write to SC_WORKFLOW
- Directly via JDBC for all other tables

Configuration of the Getter

The following keys are expected in the hashtable when communicating with the workflow server:

Key	Description
QUEUE	The queue, default is 1
DOCREFNO	The document reference number
BNO	The bank number, 3 digits
BRANCHNO	The branch number
PRIMANOTANO	The primanota number
AMOUNTLOCAL	The amount in cent
USER	The user, default is 'Getter'
DOCPRIO	The document's priority

Key	Description
RESULT	The workflow result
SCR-F.0	The count of results, excluding the 1st, the 'workflow-result', can be 0
SCR-F.<n>	The basename for the result's feature-id n=1,…,<SCR-F.0>
SCR-R.<n>	The basename for the result's resultcode n=1,…,<SCR-F.0>
SCR-M.<n>	The basename for the result's matchrate if any, default is 0 n=1,…,<SCR
SCR-C.	The basename for the result's comment n=1,…,<SCR-F.0>

Special keys in the hashtable

The following keys must be defined after reading a record from the data file and processing the first table properties file:

Key	Description
BNO	
COUNTRYID	
BANKCODE	
CUSTOMERNO	
DOCREFNO	Document reference number
MACCOUNT	Check account
ACCOUNT	SignBase account
AMOUNT	In cent
CURRENCY	EUR, USD, ...
CLEARDATE	Clearing date
FORMTYPE	0 - normal, 3 - correction item, 4 - IRD
FRONTIMAGE	The front image
BACKIMAGE	The back image
CROP0	Crop area for the signature search
CROP1	Crop area in case of failing signature search

The following keys are expected in the hashtable when communicating with the workflow server:

Key	Description
QUEUE	The queue, default is 1
DOCREFNO	The document reference number

Key	Description
BNO	The bank number, 3 digits
BRANCHNO	The branch number
PRIMANOTANO	The primanota number
AMOUNTLOCAL	The amount in cent
USER	The user, default is 'Getter'
DOCPRIO	The document's priority
RESULT	The workflow result
SCR-F.0	The count of results, excluding the 1st, the 'workflow-result', can be 0
SCR-F.<n>	The basename for the result's feature-id n=1,…,<SCR-F.0>
SCR-R.<n>	The basename for the result's resultcode n=1,…,<SCR-F.0>
SCR-M.<n>	The basename for the result's matchrate if any, default is 0 n=1,…,<SCR-F.0>
SCR-C.	The basename for the result's comment n=1,…,<SCR-F.0>

Putter

Here, too, SignCheck has a standardized file interface (result file) in which the final result and all other results of the SignCheck process is represented. Through the SC Workflow Router, the Putter can also return the current status at any time to the payment transactions system. In addition to temporary storage in the SignCheck database, the Putter writes the results of the SignCheck verification process into a standardized file. This file is received by the associated payment transactions system as feedback in order to begin the necessary steps for debiting. A document can be "accepted" through the decision of an employee in visual inspection or through the automats alone. Depending on the security requirements and quality of the document signatures, the "tolerance" of the automats can be adjusted to a compromise appropriate for production (calibration). After the results are output with the Putter program, the document status in the SignCheck workflow is set to a value that allows no further processing.

The name of the result file depends on primanota processing or not. When primanota processing is configured, the Putter writes all result records belonging to one primanota in one result file with the same name as the Getter’s data file, but with the suffix specified with key dataSuffix. Therefore the Putter has to wait until ALL records belonging to one primanota are completely processed before writing the results.

When no primanota processing is specified, the putter creates an unique file name and stores those results in this file, that have been processed, regardless from which data file the came.

As a rule, physical storage of the file takes place on the same file server through which SignCheck receives the document files. From there, the results files are generally used for archiving and booking — and also for sorting out the "return items" in cheque-based installations.

There are 2 versions of Putter, the "classic" Putter, without support of "stored procedures", and a new version PutterSP, supporting "stored procedures", this version is significant faster than the classic version, especially when processing huge amounts of data.

Note Multiple Putter programs may be running at a time. However, you must make sure within the Putter properties, that

- one BNO is only served by one Putter
- all BNOs that you process are covered by one Putter

Alternatively you can set the key maxRows >0. Then every Putter processes at most ‚maxRows' items at once and therefore work is left over for the remaining Putters. Every Putter must have different values configured for the keys statusGlobalFinished and statusGlobalWritten. If primanota-processing is defined, this method is not applicable.

Configuration of the Putter

Additional to the above described configuration of all main configuration files there are the following keys:

Key	Default	Description
tempSuffix	„.er1“	Temporary suffix of all output files. First all results are written to a file with this suffix. After finishing and successful closing this file its suffix is renamed to dataSuffix.
outputFile(f)	a non-existent file is chosen automatically	The Putter's results are written to a file with the name according to the value of outputFile. Example `outputFile = P${|DATE”yyyyMMddHH”}` the current date is 8.11.2002 15:36 the resulting filename is: "P2015102111", followed by the value of dataSuffix. Between 11.00 and 11.59.59.999 all results are written to this file, after this period results are written to the file "P2015102112" etc.
trailers	0	Number of records that have to be appended to the result file when closing
trailer1(f)		First record to be appended
trailer2(f)		Second record to be appended etc.
writeEOF	false	If set to 1, the character EOF (hex 1A) is written at the end of the file
writeCustomerType	0	If set to true, then the according CUSTOMER entry is read for getting the customer type
extraSelection	no extra selection	A where-condition indicating that the Putter is supposed to process only that part of the table SC_INTERFACE, that is covered by this where-condition, e.g. “BNO='002'”

Key	Default	Description
showExtraSelection	false	Shall the where-condition in extraSelection be shown on the GUI and be changeable?
extraLabel	"additional Selection"	The label for the extraSelection input field on the GUI
dataDirs	0	If the output directory has to be changed in dependency of any values from table SC_INTERFACE, this key defines the number of different output directories. The names of the keys for selecting these directories are dataDir1, dataDir2, etc. If primanota processing is enabled, the output directory is calculated from the values of the first row of a primanota, that will be processed. This directory will not change for a primanota, even if subsequent records of this primanota required a different directory. If primanota processing is disabled, every change of the output directory entails the creation of a new output file. If more than one definition applies to a record, the first definition that applies, is taken If none of the definitions applies, the output directory defaults to the value in the key dataDir.

Key	Default	Description

dataDir1		Definition of the conditions for the first output directory.
		Syntax
		`<column><compare-operator><value>[<logical-operator><column><compare-operator><value>[…]]=<directory>`
		where:
		`column`
		the name of a column of table SC_INTERFACE. The content of this column is compared with value.
		`compare-operator`
		one of the operators >, >=, <, <=, ==, !=. The comparison is made numerically, if both operands are numerical, otherwise lexicographically.
		`value`
		the value to compare with the value from the column above
		logical-operator
		one of the operators | (logical or) and & (logical and). These operators are executed from left to right without any precedence
		`directory`
		the output directory that is to be used, if all conditions apply
		Example
		dataDir1=BNO>001&BNO<005|PRIMANOTA_NO>=300000=dir1
		This means, if BNO is 002, 003 or 004 or PRIMANOTA_NO is greater or equals 300000, the output directory will be dir1.

dataDir2		Definition of the conditions for the first output directory etc.
statusGlobalCol		Name of the column in the table SC_INTERFACE containing the status of processing.
statusGlobalFinished	7	Value in the column statusGlobalCol indicating that the processing of this item has been finished. If multiple Putters are planned to run simultaneously, every Putter must define a different value.
statusGlobalWritten	8	Value in the column statusGlobalCol indicating that this item has been written to the result file. If multiple Putters are planned to run simultaneously, every Putter must define a different value.
statusGlobalClosed	9	Value in the column statusGlobalCol indicating that the result file is closed and renamed, i.e. this record could be removed by DFP.

Key	Default	Description
updateStatusSelect		SQL command for getting all documents that have been written
updateStatusWhere		Where clause for SQL commands to get rows from other tables for the above documents.
maxRows	0	If maxRows >0, then only this number of records are written into a result file (or less if not enough available). This setting also allows the use of multiple Putters simultaneously, but each of these Putters must define different values for statusGlobalFinished and statusGlobalWritten. If primanotaTableResource is defined, the use of multiple Putters is not possible.
useSQLCursor	true	Shall the Putter work with cursors instead of where-clauses? Cursors are faster, but not available in all database system's JDBC.
orderBy		ORDER BY clause for the SELECT statement in the first table properties file
notEqualToken	!=	Token for the not equal operand
onlyOneInstance	false	If true then only one instance of Putter can run.

commentResource		Name of the properties file containing the comments of all return codes of all SC-queues in the form:
		`Queue_Code = Text`
		where
		`Queue`
		the name of the queue
		`Code`
		the return code from this queue
		`Text`
		the comment to this return code

primanotaTableResource	no Primanota-Processing	Name of the properties file that describes the primanota-table. The next 4 entries are ignored if this entry is empty.
primanotaNamePN		Name of the column in the primanota table containing the primanota name
primanotaNameIF		Name of the column in the table SC_INTERFACE containing the primanota name
primanotaFilenameCol		Name of the column in the primanota-table containing the filename of the Getter input file without path and extension.
primanotaItemCount		Name of the column in the primanota-table containing the count of items in this primanota.
createEmptyResultFile	false	If true, a result file is created even if there is no result.
writeCRLF	true	true - write CRLF at the end of every record

Key	Default	Description
mergedFile		Name of the file that is supposed to contain all merged output files. The Putter creates this file (if specified) with the merged contents of all output files. This happens only when the Putter stops. The output files are deleted after merging.
merge.forceEODFile		Name of the file to force the end of day processing, i.e. regardless whether all items in SignCheck are processed the merging starts
merge.EODFile.0	0	Count of files that indicate with their existence the end of day, i.e. there will be no more Getter input files
merge.EODFile.1		Name of the 1st file indicating the end of day, i.e. there will be no more Getter input files
merge.EODFile.2		Name of the 2nd file indicating the end of day etc.
select.BNO		BNO for checking the database whether SignCheck processing is complete or not. Default: all BNO's (empty)
merge.GetterName	Getter	Name of the Getters main resource file, this is needed to check if all Getter files are processed

Configuration of the PutterSP

Additional to the keys described for the Putter's main configuration file there are the following keys:

Key	Default	Description
resetTableResource		Name of the properties file that performs the reset of items that could not be finally processed by a Putter, this is done only once after the start of the Putter.
prepareTableResource		Name of the properties file that performs the preparation of items finished by the workflow for the Putter, this is done once per result file.
finishTableResource		Name of the properties file that performs a flagging of the items written to a result file as finished by the Putter, this is done after closing and renaming the result file.

DFP (Day's Final Processing)

"Day's final processing" (DFP) is the final component of a processing cycle (generally the final processing at day's end), which empties the SC Database tables so that a new processing cycle can begin with the

load phase. There are 2 versions of DFP, the "classic" DFP, without support of "stored procedures", but with a statistic about the decisions to the documents and deletion depending of the final result, and a new version DFPSP, supporting "stored procedures", but without any statistic and result-independent deletion, this version is significant faster than the classic version, especially when processing huge amounts of data.

Configuration of the DFP

Additional to the above described configuration of all main configuration files there are the following keys:

Key	Default	Description
useSQLCursor	yes	Shall the DFP work with cursors instead of where-clauses? Cursors are faster, but not available in all database system's JDBC.
itemAge	"days(current date) - days(TIME_STAMP)"	The SQL expression to calculate the age in days of a record in table SC_INTERFACE.
procAge	"days(current date) - days(PROC_DATE)"	The SQL expression to calculate the age in days of the processing date of a record in table SC_INTERFACE.
pnAge	"days(current date) - days(TIME_STAMP)"	The SQL expression to calculate the age in days of an item in the primanota table.
finalResultCode2	2	The value of the FINAL_RESULT that indicates that this record is not to delete during normal (not extended) deletion.
statusGlobalCol		Name of the column in the table SC_INTERFACE containing the status of processing.
finalResultCol		Name of the column in the table SC_INTERFACE containing the FINAL_RESULT. This can be a virtual column.
statusGlobalFinished	9	Value in the column statusGlobalCol indicating that the result file is closed and renamed, i.e. this record could be removed by DFP.
dialogResource		Name of the properties file containing the settings for "Extended Deletion".
primanotaTableResource		Name of the properties file that describes the primanota-table.
Deletion of files		
deleteFiles	false	If true, additional to DFP's actual work, files are deleted.

Key	Default	Description
deleteFilesFirst	false	If true, files are deleted before deleting any database tables, otherwise files are deleted after the database work.
delPath.0	0	Is the number of directories where files are to be deleted. The directories are named in the keys delPath.1, delPath.2 etc., analog the regular expressions and ages of the files.
delPath.1	leer	The first directory where files are to be deleted. If delPath.1 is empty, nothing will be deleted.
delPattern.1	leer	Regular expression to denote the files in directory delPath.1. If delPattern.1 leer, is empty, nothing will be deleted. Example `delPattern.1=\\.done$` removes all files in the directory ending with ".done".
delDays.1	0	Only those files are deleted, that are at least ‚delDays.1' days old.
DWH2SBResource	de.softpro.signplus.service.DWH2SB	Name of the main resource file of the DWH2SB program
callDWH2SB	true	1 - start the DWH2SB program to update the SignBase database with hit rates of signatures and stockimages 0 - don't start
DWH.startScript	true	1 - start the 2 data warehouse script to store the daily production 0 - don't start scripts
DWH.script1		Name of the 1st data warehouse script to store the daily production
DWH.script2		Name of the 2nd data warehouse script to store the daily production
DWH.ignoreErrorStage1	false	true - ignore errors from DWH script 1 false - stop with error
DWH.dfpkeyResourceSB		Name of the resource file describing the access to the SignBase DFPKEY table

Key	Default	Description
DWH.dfpkeyResourceSC		Name of the resource file describing the access to the SignCheck DFPKEY table
DWH.dateCountResource		Name of the resource file describing the access to the SignCheck table to calculate the count of days
DWH.useCurrentDate	false	true - use the current date as next date false - use the date +1 of the last successful run of the DWH scripts as next date

Configuration of the DWH2SB program

This program can be started from within DFP, but can also run standalone. It updates the Hit counters of some SignBase Tables to weight the rows after their usability. This helps when removing unnecessary rows.

- Also it loads check serial number ranges into the SignBase. These ranges can be:
- The actual “issued” check serial numbers. These are notified explicitly by the bank (and loaded into the SignBase reference database).

“Observed” check serial number ranges that have been constructed by analysis of the checks accepted using SignCheck in the past. The ranges are constructed by analyzing the data from the Data Warehouse short term archive and these are loaded into the SignBase reference database by the DFP.

Analysis of the Data Warehouse short term archive also provides the list of “used” check serial numbers that is used to implement the duplicate serial number test that is part of the check serial number verification. The used status information is also loaded into the SignBase reference database by the DFP.

As well this program loads check amount statistics into the SignBase ReferenceStatistics table and loads verification of check velocity (numbers of checks processed per cycle/interval).

Additional to the above described configuration of all main configuration files there are the following keys:

Key	Default	Description
tableResourceDWH1	empty	Name of the 1st properties file for a Data Warehouse table corresponding with tableResource1
tableResourceDWH2	empty	Name of the 2nd properties file for a Data Warehouse table corresponding with tableResource2 and so on
tableResourceReset1	empty	Name of the 1st properties file for resetting the Hit Counter of a SignBase table
tableResourceReset2	empty	Name of the 2nd properties file for for resetting the Hit Counter of a SignBase table and so on
DWHResetCounters	true	If true, Hit Counter of some SignBase tables are reset, otherwise not

Key	Default	Description
tableResourceDWHSerialNumber	empty	Name of the properties file for check serial number ranges loading from Data Warehouse table
tableResourceDWHSBStockImage	empty	Name of the properties file for check serial number ranges loading to SignBase table
tableResourceDWHSBAccount	empty	Name of the properties file for check serial number ranges loading to SignBase table
tableResourceDWHArsa	empty	Name of the properties file for check amount statistics loading from Data Warehouse table
tableResourceDWHReferenceStatistics	empty	Name of the properties file for check amount statistics loading into the SignBase ReferenceStatistics table
tableResourceDWHArsv	empty	Name of the properties file for check throughput volume statistics loading into the SignBase ReferenceStatistics table
observedRangeWidth	1	The "width" of a range creation window for observed serial numbers. This is used when the DFP service program is trying to calculate the ranges that it will store into the STOCKIMAGE table. Example The input from the table vmr.arsn (for one account) contains the serial numbers: 10, 11, 21, 35, 60 If observedRangeWidth = 1, the ranges needed would be 9-12, 20-22, 34-36, 59-61 If observedRangeWidth = 5, the ranges needed would be 5-26, 30-40, 55-65 If observedRangeWidth = 10, the ranges needed would be 0-45, 50-70
observedRangeEnable	true	Flag the enables or disables the "observed" range feature. True = enable False = disable

Configuration of the DFPSP

Additional to the keys described for the DFP's main configuration file there are the following keys:

Key	Default	Description
selectBNO	false	Shall the data be deleted from the database, with a selected BNO? If Yes, the BNO label will be created on the DFP interface.

Key	Default	Description
selectBNOValue	no	If the startNow set to yes, there is the possibility to set the BNO No. directly in the property file. Works if selectBNO=true
selectBNOLabel	‘BNO’	The name of label. If selectBNO=true

Configuration of the DFP Extended Deletion Dialog

Key	Default	Description
Days	0	-1 - all entries are deleted >0 - all processed entries older than Days are deleted 0 - all processed entries are deleted, however, dependent on WaitForFR2 and DelProcDate
WaitForFR2	no	Yes - records with FINAL_RESULT=2 are never considered to be processed. These records can only be deleted with Days!=0 no - FINAL_RESULT is ignored
DelProcDate	-1	>=0 - records with PROC_DATE defined are considered to be processed only if PROC_DATE + DelProcDate days is not in the future. -1 - PROC_DATE is ignored
dialogTitle	"Extended Deletion"	Title of the “Extended Deletion Dialog”
dialogMsg	"Please select the type of Extended Deletion"	Label of the dialog message on top of the dialog
dialogButtons	" OK ,Cancel"	Names of the buttons for OK and Cancel
dialogNone	"no Extended Deletion"	Label of the "no Extended Deletion" radio button
dialogAll	"Deletion of ALL entries"	Label of the "Delete All" radio button
dialogNumber	"Deletion of entries older than"	Label of the "Delete older than" radio button
dialogDays	"days"	Label of “days” (can be language specific)
confirmTitle	"Confirm Extended Deletion"	Title of the confirm message box
confirmAll	"Are You sure to delete ALL rows?"	Message of the confirm message box for "Delete All"

Key	Default	Description
confirmNumber	"Are You sure to delete all entries older than d days?"	Message of the confirm message box for "Delete older than". The text „d“ will be replaced by the real number of days
confirmIcon	"images/dbdelete.jpg"	Location if the confirm icon
confirmButtons	"Yes,No,Cancel"	Labels of the buttons of the confirm message box
statusAll	"all entries"	The text of the status bar in case of "Delete All" is „dialogTitle: statusAll“
statusNumber	"all entries older than d days"	The text of the status bar in case of "Delete older than " is „dialogTitle: statusNumber“, the text „d“ will be replaced by the real number of days

ResultLoader

The ResultLoader is a program based on a Java SignBase client that works like a SignCheck automat: it works on a queue and tries to decide all items in this queue based on files in the data directory. This happens in batch mode, like AccountLoader or ImageLoader.

Configuration of the ResultLoader

Additional to the above described configuration of all main configuration files there are the following keys:

Key	Default	Description
user	softpro	The user name for logon to the SignBase server
password		The password for logon to the SignBase server
tempSuffix	.unprocessed	Suffix for the file containing all unprocessed records of a data file. The name of the file is the same as the data file. Unprocessed records are those records, where the according docrefno could not be found in the specified queue.
reconnectWait	10	The number of seconds to wait before trying to reconnect to the server, if this connection was lost.
queue(f)	0	Queue number where the ResultLoader works on

Key	Default	Description
defaultQueue	queue	Name of the default queue to be used when deciding all remaining documents in the queue where no suitable record in one of the data files exists.
stopTime.0	0	Count of stop times
stopTime		Default stop time. At this time the ResultLoader stops. Format is "HH:mm". The value of midnightDelay is taken into account.
stopTime.1	stopTime	1st stop time
stopTime.2	stopTime	2nd stop time etc.
midnightDelay	0	Defines the amount of minutes after midnight that are supposed to count to the previous day if midnightDelay is >0 and the amount of minutes before midnight that are supposed to count to the next day if midnightDelay is <0. midnightDelay applies for the start time and for all stop times.
midnightDelay.1	midnightDelay	midnightDelay value for the 1st stop time
midnightDelay.2	midnightDelay	midnightDelay value for the 2nd stop time etc.
EODName(f)	empty	Regular expression of an End Of Day file, if this file exists in the dataDir, the ResultLoader decides all undecided documents in the queue and stops.
defaultResult	0	Default result for the ResultLoader when deciding all undecided documents in the queue.
defaultComment	empty	Default comment for the ResultLoader when deciding all undecided documents in the queue.
defaultFeatureCode	0	Default feature code for the ResultLoader when deciding all undecided documents in the queue.
BNO	the current BNO of SignBase	The BNO to be used
holdStatus	false	Determines which documents in the queue are to be processed.
dataFiles	0	Count of the data files to be processed. If this amount of files are processed, the ResultLoader decides all undecided documents in the queue and stops.

Key	Default	Description
startTime	empty	Start time (delayed start) for the ResultLoader. If defined, then ResultLoader waits until this time is reached.
startFile.0	0	Count of start files. If >0 then ResultLoader waits until one of these files exist.
startFile.1	empty	Name of the 1st start file
startFile.2	empty	Name of the 2nd start file etc.
grouping	false	If true, all features of the same docrefno are sent to the server together, otherwise one by one.
decideRemainingItems	true	If true, the ResultLoader decides all undecided documents in the queue, otherwise not.

Special keys in the hashtable

The following keys must be defined after reading a record from the data file and processing all table properties files:

Key	Description
DOCREFNO	Contains the document reference number
FEATURECODE.0	Contains the amount of feature codes for the document
FEATURECODE.*	The feature codes for the document `*=1,…,<FEATURECODE.0>`
RESULT.*	The results of the feature codes for the document `*=1,…,<FEATURECODE.0>`
COMMENT.*	The comments of the feature codes for the document `*=1,…,<FEATURECODE.0>`
BNO	The BNO to be used, default is the resource key BNO
IGNORE	1 - the current document is ignored 0 - the current document will be processed

ResultWriter

The ResultWriter is a program based on a Java SignBase client that works like a SignCheck automat: it works on a queue and decides all items in this queue and writes 1 line per document into one or more result files. This happens in batch mode, like AccountLoader or ImageLoader.

Configuration of the ResultWriter

Additional to the above described configuration of all main configuration files there are the following keys:

Key	Default	Description
user	softpro	The user name for logon to the SignBase server
password		The password for logon to the SignBase server
reconnectWait	10	The number of seconds to wait before trying to reconnect to the server, if this connection was lost
queue(f)	0	Queue number where the ResultWriter works on
EODName(f)	empty	Name of a End Of Day file, if this file exists in the dataDir, the ResultWriter merges all written files into one file and stops
stopTime.0	0	Count of stop times
stopTime		Default stop time. At this time the ResultWriter merges all written files into one file and stops. Format is “HH:mm”. The value of midnightDelay is taken into account.
stopTime.1	stopTime	1st stop time
stopTime.2	stopTime	2nd stop time etc.
stopTimeFinish	true	Default value for stopTimeFinish.*
stopTimeFinish.1	stopTimeFinish	If true, no more documents are processed after reaching the 1st stop time
stopTimeFinish.2	stopTimeFinish	If true, no more documents are processed after reaching the 2nd stop time
midnightDelay	0	Defines the amount of minutes after midnight that are supposed to count to the previous day if midnightDelay is >0 and the amount of minutes before midnight that are supposed to count to the next day if midnightDelay is <0. midnightDelay applies for the start time and for all stop times
midnightDelay.1	midnightDelay	midnightDelay value for the 1st stop time
midnightDelay.2	midnightDelay	midnightDelay value for the 2nd stop time etc.

Key	Default	Description
writeOutput	true	If true, results are written to files, otherwise not
outputFileName	SCResultWriter.out if mergeResults=true, otherwise empty	Default output file name
outputFileName.1	outputFileName	Name of the output file for the 1st stop time
outputFileName.2	outputFileName	Name of the output file for the 2nd stop time etc.
defaultResult	0	Result for the ResultWriter when deciding a document
defaultComment	empty	Comment for the ResultWriter when deciding a document
defaultFeatureCode	0	Feature code for the ResultWriter when deciding a document
BNO	the current BNO of SignBase	The BNO to be used
holdStatus	false	Determines which documents in the queue are to be processed
mergeResults	false	If true, all result files are merged into the convenient output file
header(f)	empty	Header record for the merged result file
trailer(f)	empty	Trailer record for the merged result file
maxLineLength	0	If >0 the maximum line length, longer records are cut
readItemData	false	If true document data are read from the SignCheck database according the value of flagsItemData
flagsItemData	1	Flag for Message52 to read document data
readItemDecisions	false	If true document decisions are read from the SignCheck database
readItemReference	false	If true the document’s customer data aree read from the SignBase database
projectWriterClass	empty	Name of a class that processes every output record. In this case it makes sense to set writeOutput to false

Special keys in the hashtable

Key	Description
The following keys will be defined by the program during processing:	
DOCREFNO	Contains the current document reference number, from Message 51
PREVIOUSRESULT	Contains the previous result of the previous decision of this document, from Message 51
RC	Contains the result for this document, from Message 51
COMMENT	Contains the comment of this document, from Message 51
BNO	The BNO to be used, default is the resource key BNO
QUEUE	The queue the ResultWriter works on
SCR-F.0	The count of features codes to be used (0)
RECORD	Contains the current line when merging output files (useful for resource key “header”)
RECORDS	Contains the count of lines when merging output files (useful for resource key “trailer”)
EODFILE	Name of the EOD file, if exists
The following keys will be defined by the program during processing if resource key “readItemData”=true and in dependency of the setting of resource key “flagsItemData”:	
flagsItemData=0x0001	
ITEM.DOC_REF_NO	Document reference number
ITEM.BNO	BNO
ITEM.COUNTRYID	Country Id
ITEM.CUSTOMERNO	Customer number
ITEM.PRIMANOTA_NO	Alternative forms id number
ITEM.REL_NO_IN_PN	Together with the relative number in primanota
ITEM.SCANNER_ID	Identification of the scanner
ITEM.CHECK_ITEM	Run this through SignCheck or not
ITEM.ITEM_SOURCE	Comment field on the item
ITEM.PN_DATE	Date the primanota was put together
ITEM.SCAN_DATE	date the item was scanned
ITEM.PROC_DATE	Date the item shall be processed
ITEM.VAR_BATCH_LOAD	Flag if the item is used for a batch variant load
ITEM.DOC_PRIO	Document priority
ITEM.SERIAL_NO	Cheque serial number

Key	Description
ITEM.BANKCODE	Bank code
ITEM.BRANCH_NO	Branch sort code
ITEM.PRES_BANKCODE	Bank code of the presenting bank
ITEM.PRES_BRANCH_NO	Branch sort code of the presenting bank
ITEM.ACCOUNT	Account-Number
ITEM.M_ACCOUNT	Account-Number from MICR
ITEM.CURRENCY	Currency for llAmount
ITEM.AMOUNT	Amount
ITEM.CURRENCY_LOCAL	Currency the amount was converted to
ITEM.AMOUNT_LOCAL	Transaction amount
ITEM.SIGN_DATE	Signing date
ITEM.CLEAR_DATE	Clearing date
ITEM.C_TXN	Cheque transaction code
ITEM.FORM_TYPE	For giro forms processing
ITEM.FORM_TEXT_CODE	For giro forms
ITEM.STATUS_PUTTER	
ITEM.ID_SIGNO1	Signature id of the 1st signature
ITEM.ID_SIGNO2	Signature id of the 2nd signature
ITEM.STATUS_SCA1	
ITEM.STATUS_SCA2	
ITEM.STATUS_SCM1	
ITEM.STATUS_SCM2	
ITEM.TIME_STAMP	
flagsItemData=0x002A	
ITEM.RESOLUTION0	X resolution of the front image mono
ITEM.RESOLUTION1	Y resolution of the front image mono
ITEM.IMAGE_M0	Front image mono
flagsItemData=0x004A	
ITEM.RESOLUTION0	X resolution of the front image gray
ITEM.RESOLUTION1	Y resolution of the front image gray
ITEM.IMAGE_G0	Front image gray
flagsItemData=0x002C	
ITEM.RESOLUTION2	X resolution of the back image mono
ITEM.RESOLUTION3	Y resolution of the back image mono

Key	Description
ITEM.IMAGE_M1	Back image mono
flagsItemData=0x004C	
ITEM.RESOLUTION2	X resolution of the back image gray
ITEM.RESOLUTION3	Y resolution of the back image gray
ITEM.IMAGE_G1	Back image gray
flagsItemData=0x0032	
ITEM.SIG0_0	Left edge of the 1st signature mono
ITEM.SIG0_1	Upper edge of the 1st signature mono
ITEM.SIG0_2	Right edge of the 1st signature mono
ITEM.SIG0_3	Bottom edge of the 1st signature mono
ITEM.SIGNATURE_M0	1st signature mono
ITEM.SIG1_0	Left edge of the 2nd signature mono
ITEM.SIG1_1	Upper edge of the 2nd signature mono
ITEM.SIG1_2	Right edge of the 2nd signature mono
ITEM.SIG1_3	Bottom edge of the 2nd signature mono
ITEM.SIGNATURE_M1	2nd signature mono
flagsItemData=0x0034	
ITEM.SIG2_0	Left edge of the 3rd signature mono
ITEM.SIG2_1	Upper edge of the 3rd signature mono
ITEM.SIG2_2	Right edge of the 3rd signature mono
ITEM.SIG2_3	Bottom edge of the 3rd signature mono
ITEM.SIGNATURE_M2	3rd signature mono
flagsItemData=0x0052	
ITEM.SIG0_0	Left edge of the 1st signature gray
ITEM.SIG0_1	Upper edge of the 1st signature gray
ITEM.SIG0_2	Right edge of the 1st signature gray
ITEM.SIG0_3	Bottom edge of the 1st signature gray
ITEM.SIGNATURE_G0	1st signature gray
ITEM.SIG1_0	Left edge of the 2nd signature gray
ITEM.SIG1_1	Upper edge of the 2nd signature gray
ITEM.SIG1_2	Right edge of the 2nd signature gray
ITEM.SIG1_3	Bottom edge of the 2nd signature gray
ITEM.SIGNATURE_G1	2nd signature gray
flagsItemData=0x0054	

Key	Description
ITEM.SIG2_0	Left edge of the 3rd signature gray
ITEM.SIG2_1	Upper edge of the 3rd signature gray
ITEM.SIG2_2	Right edge of the 3rd signature gray
ITEM.SIG2_3	Bottom edge of the 3rd signature gray
ITEM.SIGNATURE_G2	3rd signature gray
The following keys will be defined by the program during processing if resource key “readItemDecisions”=true:	
DECISION.0	Count of decisions (features) already mady to this document
DECISION.F<fid>MatchRate	Match rate (0,…,100) of feature <fid>, where <fid> is one of the features defined for this document
DECISION.F<fid>Result	Result of feature <fid>, where <fid> is one of the features defined for this document
DECISION.F<fid>Matchcode	Match code (2 characters) of feature <fid>, where <fid> is one of the features defined for this document
DECISION.F<fid>Text	Comment of feature <fid>, where <fid> is one of the features defined for this document
The following keys will be defined by the program during processing if resource key “readItemReference”=true:	
REFERENCE.VALUEDC	1 if the customer is a valued one, otherwise 0
REFERENCE.CUSTOMERTYPE	The customer’s customer type (1 character)
REFERENCE.EXT0.0	Count of customer extensions
REFERENCE.EXT0.<fid>	Value of field id <fid>, where <fid> is one of the defined fields of the customer extension
REFERENCE.EXT1.0	Count of account extensions
REFERENCE.EXT1.<fid>	Value of field id <fid>, where <fid> is one of the defined fields of the account extension
The following keys can be set during processing the table resource files:	
SCR-F.0	The count of features codes to be used
SCR-F.*	The features codes to be used. *=1,…,SCR-F.0
SCR-R.*	The results for the features to be used. *=1,…,SCR-F.0
SCR-C.*	The comments for the features to be used. *=1,…,SCR-F.0

PasswordEncoder

AccountLoader, Getter, Putter and DFP need a valid userid/password combination for accessing the SignBase/SignCheck database. Also the SignatureReferenceFilter needs userid/password for accessing the SignBase server.

Userid and password are usually stored in .ini or .properties files. This has the disadvantage, that non-authorised persons have possibly access to this information. To avoid this, it is possible to store the

password encrypted. All programs that work with passwords, try the login first with the original values. If this fails, the program assumes, that the password is encrypted, decrypts the password (where a SignPlus license is needed) and tries the login with the decrypted password. Only if this failed also, the user is prompted to put in a valid userid/password combination.

There is a small program, called PasswordEncoder, which can be used for encrypting a password. The encrypted password should replace the original password in the .ini or .properties files.

Encrypting and decrypting must take place using a SignPlus license.

TableAccess

This service program is a very generic program. It allows access to a database for reading, writing etc. An input file is not expected. The normal way to use this program is to execute a query on a database table and then to process the results one by one.

Configuration of the TableAccess program

Additional to the above described configuration of all main configuration files there are the following keys:

Key	Default	Description
tableResource<n>.0	0	Count of subtables for tableResource<n>, where `n=1,…,<tables>` On the table defined in tableResource<n> is always a SELECT performed. The result set of this SELECT is the input for processing tableResource<n> again, this time depending on CONTROL.ACTION, and all subtables, row by row.
tableResource<n>.1		1st subtable for tableResource<n>, where `n=1,…,<tables>`
tableResource<n>.2		2nd subtable for tableResource<n>, where `n=1,…,<tables>`

The configuration file service.properties

For all service programs there are a lot of properties to keep these programs flexible. Those properties, that are editable by the user, are located in a special resource file named service.properties. Entries within this file have precedence against entries with the same name in other resource files!

To decide to which program and which resource file an entry belongs, every entry is preceded by a prefix. The linkage between these entries and the resource file that they belong to is made by the following key in the respective resource file:

```
$#INSERT<ppp> service:<prefix>
```

Where <ppp> is the priority of this statement and <prefix> is the prefix, e.g. Getter or Putter

Example

```
Getter.dataDir=..\\data
```

```
Putter.dataDir=..\\out
```

Defines the data directory for the Getter with

```
"..\\data"
```

but for the Putter with

```
"..\\out"
```

Overview over properties that are intended to be editable by the user

This list can be extended arbitrarily.

Program	Key	Description
AccountLoader	AL.dataDir	Initial name of the data directory
AccountLoader	AL.logFile	Initial name of the log file
AccountLoader	AL.dataSuffix	Suffix of all data files
AccountLoader	AL.errorSuffix	Suffix for the error file. When the processing of the data file fails, the log is written to the file with the name of the data file and this extension.
AccountLoader	AL.renameSuffix	Suffix for renaming the data file. When specified, after processing the data file's dataSuffix is replaced by renameSuffix.
AccountLoader	AL.rejectSuffix	If rejectSuffix is not empty, then all records that could not be properly processed are written to a reject file. This file has the name of the data file and the extension ".rejectSuffix".
AccountLoader	AL.workSuffix	Suffix for working files. The data file is locked by creating an empty working file in the same directory, with the same filename, but with the extension workSuffix. This allows more than one program to run on the same data directory without processing the same file twice.
AccountLoader	AL.REPORT.ReportPath	Path of the report files
AccountLoader	AL.saveLog	1 - the log is saved into a logfile 0 - the log is not saved
AccountLoader	AL.traceLevel	Trace level

Program	Key	Description
AccountLoader	AL.startNow	Shall the start button be pressed automatically after the program invocation? [yes|no]
AccountLoader	AL.wait	Number of seconds to wait after processing all data files before searching for new data files
AccountLoader	AL.deleteDataFile	If set to true, the successfully processed data file is not renamed, but deleted. If an activation file exists, it is deleted.
AccountLoader	AL.ruleCreate	If set to true, then create a RULE object if creating a signatory object.
AccountLoader	AL.rulePower	The POWER for creating a RULE object Default: S
ImageLoader	IL.dataDir	Initial name of the data directory
ImageLoader	IL.logFile	Initial name of the log file
ImageLoader	IL.dataSuffix	Suffix of all data files
ImageLoader	IL.errorSuffix	Suffix for the error file. When the processing of the data file fails, the log is written to the file with the name of the data file and this extension.
ImageLoader	IL.renameSuffix	Suffix for the renaming the data file. When specified, after processing the data file's dataSuffix is replaced by renameSuffix.
ImageLoader	IL.rejectSuffix	If rejectSuffix is not empty, then all records that could not be properly processed are written to a reject file. This file has the name of the data file and the extension „.rejectSuffix“
ImageLoader	IL.workSuffix	Suffix for working files. The data file is locked by creating an empty working file in the same directory, with the same filename, but with the extension workSuffix. This allows more than one program to run on the same data directory without processing the same file twice.
ImageLoader	IL.REPORT.ReportPath	Path of the report files
ImageLoader	IL.saveLog	1 - the log is saved into a logfile 0 - the log is not saved
ImageLoader	IL.traceLevel	Trace level

Program	Key	Description
ImageLoader	IL.startNow	Shall the start button be pressed automatically after the program invocation? [yes|no]
ImageLoader	IL.wait	Number of seconds to wait after processing all data files before searching for new data files
ImageLoader	IL.deleteDataFile	If set to true, the successfully processed data file is not renamed, but deleted. If an activation file exists, it is deleted.
ImageLoader	IL.check.AmountRange	1 - amount range check 0 - no amount range check
ImageLoader	IL.ignoreRange.0	Count of amount ignore ranges
ImageLoader	IL.ignoreRange.1	1st ignore range. Format: “<minAmount>,<maxAmount>”, all amounts in cent.
ImageLoader	IL.ignoreRangePrivate.0	Count of amount ignore ranges for private accounts
ImageLoader	IL.ignoreRangePrivate.1	1st ignore range for private accounts
ImageLoader	IL.ignoreRangeCorporate.0	Count of amount ignore ranges for corporate accounts
ImageLoader	IL.ignoreRangeCorporate.1	1st ignore range for corporate accounts
ImageLoader	IL.ignoreRangeCorporate.2	2nd ignore range for corporate accounts
ImageLoader	IL.ignoreRangeOther.0	Count of amount ignore ranges for other accounts
ImageLoader	IL.ignoreRangeOther.1	1st ignore range for other accounts
ImageLoader	IL.ignoreRangeOther.2	2nd ignore range for other accounts
ImageLoader	IL.check.IRDCheck	1 - IRD (Image Replacement Document) check 0 - no IRD check
ImageLoader	IL.check.CorrectionItemCheck	1 - Correction Item check 0 - no Correction Item check
ImageLoader	IL.check.UnusualSizeCheck	1 - Unusual Size check 0 - no Unusual Size check
ImageLoader	IL.check.PADCheck	1 - PAD (Pre Authorized Draft) check 0 - no PAD check
ImageLoader	IL.PADLevel	Confidence level for PAD

Program	Key	Description
ImageLoader	IL.check.prePADCheck	1 - Pre PAD check 0 - no Pre PAD check (only for performance reasons
ImageLoader	IL.check.PADCleanedSizeMin	Minimum size (in bytes) of a cleaned PAD
ImageLoader	IL.check.PADCleanedSizeMax	Maximum size (in bytes) of a cleaned PAD
ImageLoader	IL.check.ASVCheck	1 - ASV check 0 - no ASV check
ImageLoader	IL.check.SerialNoCheck	1 - Serial Number check 0 - no Serial Number check
ImageLoader	IL.check.minSerialNo	Minimum Serial Number
ImageLoader	IL.deltaSize	Maximum difference of width and height in pixels for comparing 2 images
ImageLoader	IL.FPLevel	Confidence level for checkstock compare
ImageLoader	IL.maxImagesPrivate	Maximum count of checkstock images for private accounts
ImageLoader	IL.maxImagesCorporate	Maximum count of checkstock images for corporate accounts
ImageLoader	IL.maxImagesOther	Maximum count of checkstock images for other accounts
ImageLoader	IL.minAgeOldImages	Minimum age of old images in days
ImageLoader	IL.deleteOldImages	1 - delete old images if the maximum is reached 0 - don’t delete old images
ImageLoader	IL.check.VariantsCheck	1 - check if the maximum count of variants is reached 0 - don’t check
ImageLoader	IL.check.CreateSRFData	1 - create SRF data 0 - don’t create SRF data
ImageLoader	ImageLoader.check.CustomerQuery	1 - query the customer number from the SignBase database 0 - the customer number is available
ImageLoader	IL.SRFTempSuffix	Temporary data suffix for SRF files
ImageLoader	IL.SRFDataSuffix	Final data suffix for SRF files
ImageLoader	IL.report	1 - create a report file 0 - don’t create a report file

Program	Key	Description
ImageLoader	IL.reportOnlyDB	1 - report only database changes 0 - report every record
SignatureReferenceFilter	SRF.dataDir	Initial name of the data directory
SignatureReferenceFilter	SRF.logFile	Initial name of the log file
SignatureReferenceFilter	SRF.renameSuffix	Suffix for the renaming the data file. When specified, after processing the data file's dataSuffix is replaced by renameSuffix.
SignatureReferenceFilter	SRF.saveLog	1 - the log is saved into a logfile 0 - the log is not saved
SignatureReferenceFilter	SRF.traceLevel	Trace level
SignatureReferenceFilter	SRF.startNow	Shall the start button be pressed automatically after the program invocation? [yes|no]
SignatureReferenceFilter	SRF.createCustomer	1 - create a customer object if not exists 0 - don't create a customer
SignatureReferenceFilter	SRF.createAccount	1 - create an account object if not exists 0 - don't create an account
SignatureReferenceFilter	SRF.createDummySignatory	1 - create a Dummy Signatory 0 - don't create a Dummy Signatory
SignatureReferenceFilter	SRF.assignToDummySignatory	1 - assign a new variant to the Dummy Signatory 0 - don't assign
SignatureReferenceFilter	SRF.sivalMatchRate	Confidence level for sival compare, values go from AA to
SignatureReferenceFilter	SRF.Quality.SmallSnippet	Threshold width in pixels between small and large signature snippets
SignatureReferenceFilter	SRF.Quality.SmallBBB	Maximum of black pixels in percent in the signature snippet for small snippets
SignatureReferenceFilter	SRF.Quality.LargeBBB	Maximum of black pixels in percent in the signature snippet for large snippets
SignatureReferenceFilter	SRF.Quality.SmallSimplicity	Maximum simplicity of the signature snippet for small snippets (0…100)
SignatureReferenceFilter	SRF.Quality.LargeSimplicity	Maximum simplicity of the signature snippet for large snippets (0…100)
SignatureReferenceFilter	SRF.storeIRD	1 - store variants from an IRD 0 - don't store

Program	Key	Description
FraudFeedbackFileLoader	F3.dataDir	Initial name of the data directory
FraudFeedbackFileLoader	F3.logFile	Initial name of the log file
FraudFeedbackFileLoader	F3.wait	Number of seconds to wait after processing all data files before searching for new data files
FraudFeedbackFileLoader	F3.REPORT.ReportPath	Path of the report files
FraudFeedbackFileLoader	F3.saveLog	1 - the log is saved into a logfile 0 - the log is not saved
FraudFeedbackFileLoader	F3.traceLevel	Trace level
FraudFeedbackFileLoader	F3.CountryId	CountryId to be used
FraudFeedbackFileLoader	F3.BankCode	BankCode to be used
FraudFeedbackFileLoader	F3.blockVarValidFrom	Count of days from today when the blocked variant will be valid
FraudFeedbackFileLoader	F3.unblockVarValidFrom	Count of days from today when the unblocked variant will be valid
FraudFeedbackFileLoader	F3.blockImageValidFrom	Count of days from today when the blocked stockimage will be valid
FraudFeedbackFileLoader	F3.unblockImageValidFrom	Count of days from today when the unblocked stockimage will be valid
FraudFeedbackFileLoader	F3.blockingTime	Blocking time in days for stockimages
FraudFeedbackFileLoader	F3.reportOnlyDB	1 - report only database changes 0 - report every record
Getter	Getter.dataDir	Initial name of the data directory
Getter	Getter.logFile	Initial name of the log file
Getter	Getter.dataSuffix	Suffix of all data files
Getter	Getter.renameSuffix	Suffix for renaming the data file. When specified, after processing the data file's dataSuffix is replaced by renameSuffix.
Getter	Getter.workSuffix	Suffix for working files. The data file is locked by creating an empty working file in the same directory, with the same filename, but with the extension workSuffix. This allows more than one program to run on the same data directory without processing the same file twice.
Getter	Getter.saveLog	1 - the log is saved into a logfile 0 - the log is not saved
Getter	Getter.traceLevel	Trace level

Program	Key	Description
Getter	Getter.startNow	Shall the start button be pressed automatically after the program invocation? [yes|no]
Getter	Getter.wait	Number of seconds to wait after processing all data files before searching for new data files
Getter	Getter.AccountHolderLevel	Confidence level for the PAD Accountholder check
Getter	Getter.BlacklistLevel	Confidence level for the PAD blacklist check
Getter	Getter.check.AmountRange	1 - amount range check 0 - no amount range check
Getter	Getter.ignoreRange.0	Count of amount ignore ranges
Getter	Getter.ignoreRange.1	1st ignore range. Format: "<minAmount>,<maxAmount>", all amounts in cent
Getter	Getter.check.CustomerQuery	1 - query the customer number from the SignBase database 0 - the customer number is available
Getter	Getter.check.IRDCheck	1 - IRD (Image Replacement Document) check 0 - no IRD check
Getter	Getter.check.CorrectionItemCheck	1 - Correction Item check 0 - no Correction Item check
Getter	Getter.check.UnusualSizeCheck	1 - Unusual Size check 0 - no Unusual Size check
Getter	Getter.check.PADCheck	1 - PAD (Pre Authorized Draft) check 0 - no PAD check
Getter	Getter.PADLevel	Confidence level for PAD
Getter	Getter.check.AccountCheck	1 - check for existence of the account 0 - don’t check
Getter	Getter.check.SignatureSizeCheck	1 - check if the size of the signature (in bytes) is not to big for the according database column 0 - don’t check
Getter	Getter.PADkey.0	Count of keys containing keywords for PAD
Getter	Getter.PADkey.1	1st keyword for PAD
Getter	Getter.Host	Hostname of the workflow server
Getter	Getter.Port	tcp/ip port of the workflow server

Program	Key	Description
Getter	Getter.Timeout	Timeout in seconds for messages to the workflow server
Getter	Getter.maxRetries	Maximum number of retries when the sending of data to the workflow server times out, before an error is thrown
Putter	Putter.dataDir	Initial name of the data directory
Putter	Putter.logFile	Initial name of the log file
Putter	Putter.traceLevel	Trace level
Putter	Putter.saveLog	1 - the log is saved into a logfile 0 - the log is not saved
Putter	Putter.startNow	Shall the start button be pressed automatically after the program invocation? [yes|no]
DFP	DFP.dataDir	Initial name of the data directory
DFP	DFP.logFile	Initial name of the log file
DFP	DFP.traceLevel	Trace level
DFP	DFP.saveLog	1 - the log is saved into a log file 0 - the log is not saved
DFP	DFP.startNow	Shall the start button be pressed automatically after the program invocation? [yes|no]
DFP	DFP.Days	[-1 | 0 | >0] -1 - all entries are deleted >0 - all processed entries older than 'Days' are deleted =0 - all processed entries are deleted, however, dependent on WaitForFR2 and DelProcDate
DFP	DFP.selectBNO	
DFP	DFP.deleteFiles	1 - delete files in the filesystem 0 - don’t delete files
DFP	DFP.deleteFilesFirst	1 - delete files before working on the database 0 - delete files after working on the database
DFP	DFP.delPath.0	Count of paths where files are to be deleted
DFP	DFP.delDays.1	1st minimum age in days of files to be deleted
DFP	DFP.delPath.1	1st path where files are to be deleted

Program	Key	Description
DFP	DFP.delPattern.1	1st file pattern (regular expression) of files to be deleted
DFP	DFP.callDWH2SB	1 - start the DWH2SB program to update the SignBase database with hit rates of signatures and stockimages 0 - don’t start
DFP	DFP.DWH.startScript	1 - start the 2 data warehouse scripts to store the daily production 0 - don’t start scripts
ImageLoader, SRF	Variants.maxVariants	Maximum Variants per customer and signatory
ImageLoader, SRF	Variants.maxVariantsPrivate	Maximum Variants per customer and signatory for private accounts
ImageLoader, SRF	Variants.maxVariantsCorporate	Maximum Variants per customer and signatory for corporate accounts
ImageLoader, SRF	Variants.maxVariantsOther	Maximum Variants per customer and signatory for other accounts
ImageLoader, SRF	Variants.validFrom	Count of days after the clearing date where a variant becomes valid
ImageLoader, SRF	Variants.minAgeOldVariants	Minimum age in days for old variants
ImageLoader, SRF	Variants.deleteOldVariants	1 - delete old variants when the maximum variants is reached 0 - don’t delete
Getter, SRF	Crop.clipResolution	Resolution for all cropping rectangles
Getter, SRF	Crop.CROP0Default	Rectangle of the search area for the LINESEARCH function
Getter, SRF	Crop.CROP1Default	Rectangle for signature 1 on the front side
Getter, SRF	Crop.CROP2Default	Rectangle for signature 2 on the front side
Getter, SRF	Crop.cleanLevel	Sival cleaning level in per mill
Getter, SRF	Crop.cleanLines	1 - clean lines 0 - don’t clean lines
All service programs that work with a database via JDBC	Database.URL	URL of the SignBase database
	Database.propValue1	User id for the SignBase database
	Database.propValue2	Password for the SignBase database
	Database.catalog	Catalog of the SignBase database
	DatabaseSC.URL	URL of the SignCheck database
	DatabaseSC.propValue1	User id for the SignCheck database

Program	Key	Description
	DatabaseSC.propValue2	password for the SignCheck database
	DatabaseSC.catalog	Catalog of the SignCheck database
	DatabaseDWH.URL	URL of the data warehouse database
	DatabaseDWH.propValue1	User id for the data warehouse database
	DatabaseDWH.propValue2	Password for the data warehouse database
	DatabaseDWH.catalog	Catalog of the data warehouse database

Changing the name of the configuration file service.properties

To use another configuration file than service.properties, the parameter list of the calling program must contain the parameter

```
$#USEservice=<new Name>
```

where <new Name> is the name of the file (without ".properties").

Signature selection

General

In cases, where the signature snippet is not delivered, but the whole image of the item, there is a method for defining the region of a signature.

Fix values of the signature snippet are the image format, the depth (number of bits per pixel) and a resolution of 300dpi for the coordinates of the region. If the real resolution is not 300dpi, a conversion of the coordinates occurs.

The coordinates of the rectangle can be changed. Every item type can have its own coordinates for the signature region.

Signature search

It is recommended to use the function LINESEARCH to find the signature area. Input is the whole image and a rectangle probably containing the signature. It is also possible, but time-consuming to use the whole image. LINESEARCH returns all areas that could contain a signature, in the order from lower right to upper left, because commonly a signature can be found in the lower right corner of a check.

Syntax of the definition of a rectangle

The rectangle is defined by the upper left and the lower right corner of the rectangle. The best way to define this rectangle is:

```
left,top,right,bottom
```

It is also possible to use negative values. In this case the real value is:

width of the image (or height resp.) + value

or in other words, negative values are counted from the opposite edge of the image.

You can seek for a line inside the rectangle and take the found line as new coordinate for this edge:

```
left+seekleft,top+seektop,right+seekright,bottom+seekbottom
```

where seek… is the number of pixels to seek for a line starting from the edge to the middle of the rectangle.

And you can change the position of a new edge after a successful search of a line by defining the new edge relative to the found line:

```
left+seek+|-indent,top+seek+|-indent,right+seek+|-indent,bottom+seek+|-indent
```

where indent is the distance of the new edge to the found line. A positive indent means always the direction to the middle of the rectangle.

Or you can define the width or height of the rectangle if a line was found by defining the width/height on the opposite edge:

```
left+seek+|-indent,top+seek+|-indent,right-width,bottom-height
```

or

```
left-width,top-width,right+seek+|-indent,bottom+seek+|-indent
```

or any other combination.

Note In case that the bank's clearing system is not able to identify the check item type because of missing information on the check itself, it is possible to use additional check definitions. If all relevant check types are of a different check size, it is possible to use the check size.

Chapter 3

Overview about the Java Files

See javadoc located in \\gimli\spdfs\dev\rel\core\SignPlus\R43\freeze\service\TiffSoftproServiceDoc.zip but project-specific java code is not documented there.