CA Role & Compliance Manager

Unique User ID (UUID) User Guide

r4.1.2

Contact CA

Contact Technical Support

For your convenience, CA provides one site where you can access the information you need for your Home Office, Small Business, and Enterprise CA products. At http://ca.com/support, you can access the following:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Provide Feedback

If you have comments or questions about CA product documentation, you can send a message to techpubs@ca.com.

If you would like to provide feedback about CA product documentation, complete our short customer survey, which is also available on the CA support website, found at http://ca.com/support.

CA Product References

This document references the following CA products:

- CA Role & Compliance Manager
- CA Identity Manager

Contents

Chapter 1: Introducing the UUID Utility

One of the first challenges in every identity, compliance, and roles project is consolidating all of the access rights of a given person from all systems and applications. This requires correlating diverse user IDs and account names to their rightful owners. In enterprise implementations there is often no unique naming convention for accounts, which makes the process difficult.

The Eurekify Unique User ID Correlation Tool (UUID) addresses this problem as follows:

- Provides an easy-to-use declarative tool that automatically identifies accounts based on common naming conventions where they do exist
- Uses pattern recognition technology to match accounts that do not adhere to any naming convention (such as root, or process accounts)
- Uses pattern recognition technology to remove or reduce ambiguity
- Consolidates and merges imported access rights based on agreed account matches

This section contains the following topics:

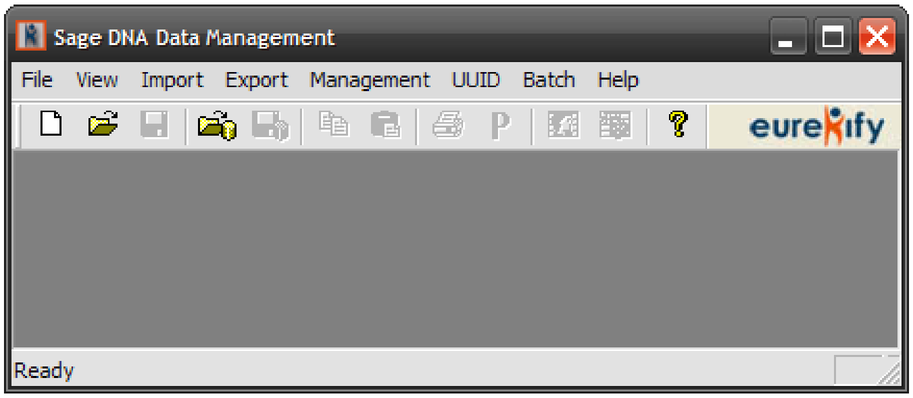
Start the UUID Utility

Start the UUID utility using the Eurekify Sage Data Manager (DM).

To start the UUID Utility

1. Click Start, All Programs, Eurekify Sage ERM, Eurekify Sage Data Manager V4.0.

   The Sage DM window opens.

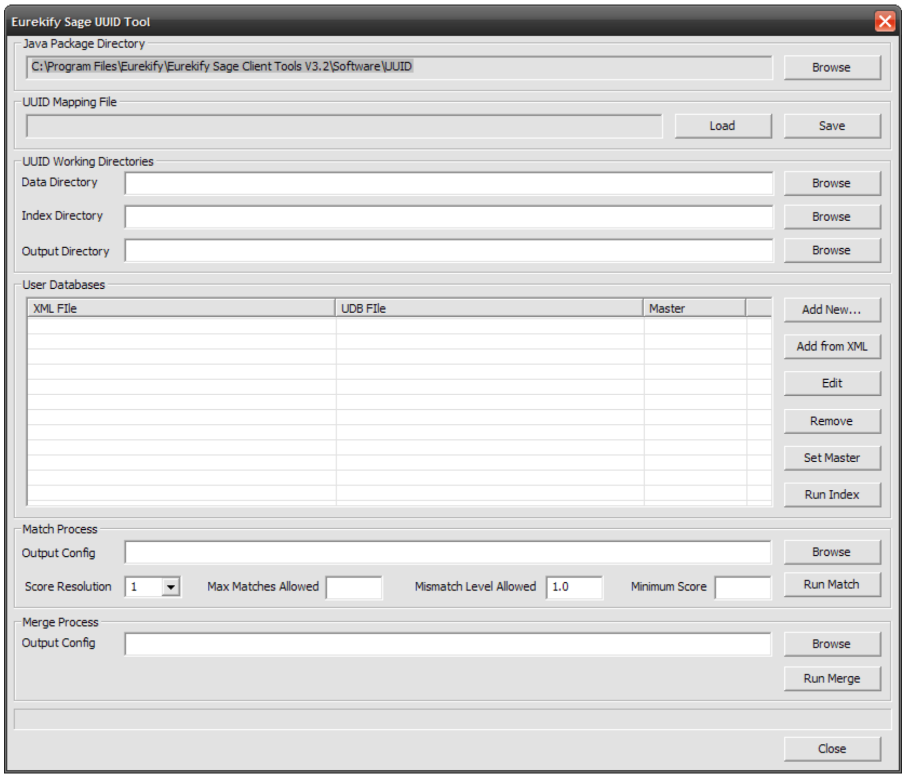


2. From the UUID menu select Launch UUID Tool.

   The UUID Tool opens. On the first occasion that you run the UUID tool, the interface opens displaying blank input fields.

The UUID Interface

The Eurekify user interface is divided into several sections that reflect the work process that you undertake in consolidating the access rights and privileges on your system. The following is a sample:



Java Package Directory: C:\Program Files\Eurekify\Eurekify Sage Client Tools V3.2\Software\UUID

Score Resolution: 1

Mismatch Level Allowed: 1.0

The following table describes the sections:

Section	Description
Java Package Directory	The path in which the UUID package is located. (this is where the EurekifyMatcher.jar is located)
UUID Mapping File	The main settings file that refers to all other definitions.
UUID Working Directories	Defines the locations in which the tool can find source data and deposit temporary output files that contain consolidated output data. (all directories here must be on same drive, for example, C:\)
User Databases	Provide mappings that map each of the accounts sources (Eurekify user databases).

Section	Description
Match Process	Provides the file name and directory of the configuration that results from the matching process, as well as a few general parameters for the matching. Also runs the process that performs the matching process.
Merge Process	Provides the file name and directory of the resulting configuration file that contains the consolidated access rights based on the above matching. Also runs the process that performs the merging process.

UUID Work Process

This section describes the general work flow that you perform when using the UUID tool.

The general work process is as follows:

1. Install Eurekify Sage DNA Data Manager and then Install the Eurekify UUID packages. The UUID software is provided in two packages: one includes the Eurekify software, and one includes open source modules on which it depends.
2. Install a license that includes the Eurekify UUID tool.
3. For each of your company systems you must extract or export the user data and save it in the form of a CSV file in the same format as a Eurekify Users DB (UDB). Each of the csv files should be renamed so that they use a *.udb extension. If you have imported the full access rights from those systems in a Eurekify configuration, you can use the UDB from these configurations. You must create a data directory and then place the *.udb, or *.cfg files in the data directory.
4. Run the UUID tool.

   Specify the path and file name for the Eurekify UUID package.
5. Specify the path for the UUID Working Directories: Data Directory, Index Directory and Output Directory. (note that all directories must be on same logical drive, e.g., C:\).
6. Define the mapping definitions for matching users to their resources and accounts across available systems and save the mapping definitions file.
7. Run the Index.
8. Enter the path and name of the configuration file that contains the matched data in the Match Process section and run the Match process.
9. If desired enter the path and name of the configuration file that contains the merged data in the Merge Process section and run the Merge process.

Installing the UUID Tool

The UUID tool's installation package can be downloaded from Eurekify. Contact Eurekify support services for download details.

To complete the installation process

1. Verify you have Java Runtime Environment version 1.5 or later installed on your machine, and that the Java BIN directory is in your system PATH.
2. Copy the installation package to your system.
3. Extract the supplied zip files to the Eurekify program directory as instructed.
4. Install the UUID license file.

UUID Installation Package

The UUID installation package includes the following components:

Folder	Contents
UUID-Eurekify	Readme Text file EurekifyMatcher.zip file UUID Demo.zip file
UUID-OpenSource	UUID_libs.zip These are open source modules on which the Eurekify UUID software depends

To install the UUID tool

1. Create a directory called UUID in the Eurekify program path as follows:

   <Install Drive>:\Program Files\Eurekify\Eurekify Sage Client Tools V[version]\Software\UUID

   where [*version*] is V3.2 or V4.0.
2. Create a directory called *lib* in the UUID directory.
3. Extract the contents of the UUID_libs.zip into the following directory:

   <Install Drive>:\Program Files\Eurekify\Eurekify Sage Client Tools V[version]\Software\UUID\lib

   where [*version*] is V3.2 or V4.0.
4. Extract the contents of the *EurekifyMatcher*.zip file to newly created *UUID* directory.
5. Access DNA Data Management, a short installation procedure for UUID is performed.

Licenses and License Renewal

Obtain from Eurekify support a license which includes permission to use the UUID tool.

See the Eurekify Sage DNA user-manual, General Settings / License and Version section for more details on how to activate the license.

Prepare Company HR and Systems Data

Using proprietary pattern recognition technology the UUID tool identifies and matches users to their accounts across all of your company systems. The source data used by the UUID tool is the user and account data for each system saved in the form of a CSV file. The format for this file is exactly the same as any other Eurekify UDB. If you have imported a full configuration from a certain system, you can simply use its UDB here.

For each of your company systems:

Copy the *.udb files (or full set of .cfg, .udb, and .rdb) to the data directory.

The Data Directory is referenced as one of the Working Directories. The UDB files are used by the UUID tool during the matching and merging process.

Set Java Package Directory

The Java Package section in the UUID Tool references the installation directory that contains the EurekifyMatcher.jar file.

To set the Java Package Directory

1. In the Java Package Directory section click *Browse*.

   A Browse dialog opens.

2. Navigate to and select the *<Install Drive>:\Program Files\Eurekify\Eurekify Sage Client Tools V[version]\Software\UUID* directory (where [*version*] is V3.2 or V4.0), and click OK.

   The selected directory appears in the text field in the Java Package Directory section.

Working Directories

The Working Directories are a set of directories on your local machine that are used to house data and deposit output files that contain consolidated output data. The Data Directory is used to store your *.udb files that contain data extracted from your various company systems.

Note: All working directories must be placed on same logical drive, such as C:\.

Working Directory	Description
Data Directory	Stores data files containing user and account data extracted from the various company systems.
Index Directory	Stores internal UUID files generated as part of the Indexing process. **Note:** Erasing or editing these files causes the UUID tool to malfunction.
Output Directory	Provides a container to house temporary output files that are for internal use by the UUID tool only. **Note:** Erasing or editing these files will cause the UUID tool to malfunction.

Create and Assign Working Directories

You need to create each of the working directories on your database server and then assign their path in the UUID tool.

To create and assign work directories

1. On your local machine, create three directories, one each for your Data Directory, Index Directory, and Output Directory.

   For example using the directory path C:\testdemo\uuid_demo, create the following directories:

   Data Directory

   C:\test\uuid_demo\demodata

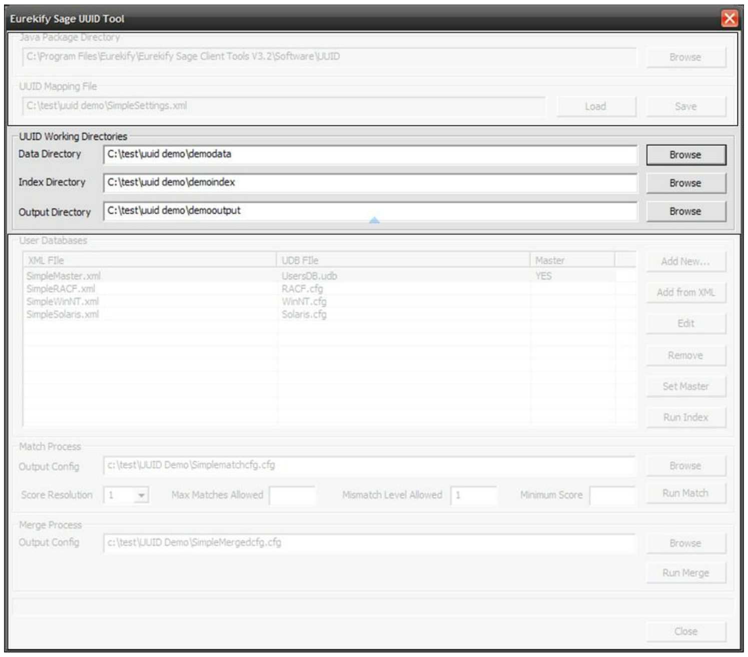
   Index Directory

   C:\test\uuid_demo\demoindex

   Output Directory

   C:\test\uuid_demo\demooutput

2. In the UUID Working Directory section of the UUID tool (highlighted in the following screen), enter the directory path in the text field for each of the directories that you created. To search for the directory click Browse.



Eurekify Sage UUID Tool

Java Package Directory: C:\Program Files\Eurekify\Eurekify Sage Client Tools V3.2\Software\UUID

UUID Mapping File: C:\test\uuid demo\SimpleSettings.xml

UUID Working Directories
Data Directory: C:\test\uuid demo\demodata
Index Directory: C:\test\uuid demo\demoindex
Output Directory: C:\test\uuid demo\demooutput

User Databases

XML FIle	UDB FIle	Master
SimpleMaster.xml	UsersDB.udb	YES
SimpleRACF.xml	RACF.cfg	
SimpleWinNT.xml	WinNT.cfg	
SimpleSolaris.xml	Solaris.cfg	

Match Process
Output Config: c:\test\UUID Demo\Simplematchcfg.cfg
Score Resolution: 1   Max Matches Allowed: Mismatch Level Allowed: 1   Minimum Score:

Merge Process
Output Config: c:\test\UUID Demo\SimpleMergedcfg.cfg

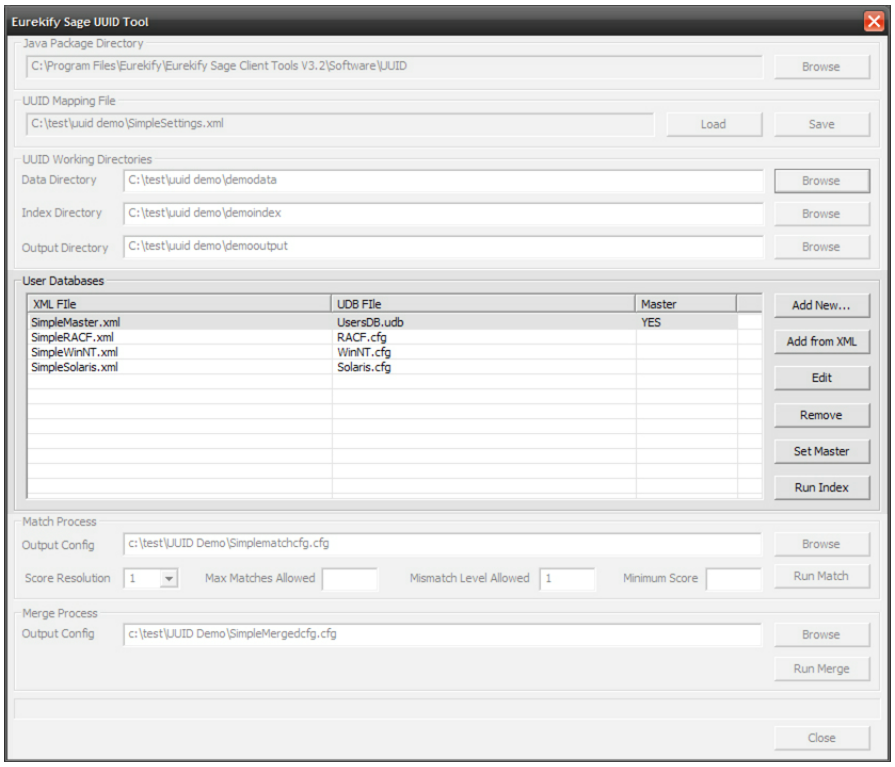
3. Select the directory, click OK.

   The directory path is displayed in the selected Working Directory text field.

User Databases

The User Databases section of the UUID Tool (highlighted in the following screen) is where you define the parameters and settings, and identify data that is used to consolidate the user access rights and privileges across all systems in your organization. Your goal is to identify each person in your organization with the accounts they have access to on each of the systems in your organization.

In some cases this is straight forward, for example, if the organization's personnel use the same account ID on all systems. In other cases, it may be possible to identify the owner of an account because accounts are based on some naming convention, for example, jdoe for John Doe. In the more difficult cases, it may be possible to recognize the account owner based on cues in some of the other account fields, for example, name (free text), address, phone number, email address, and so on. This information is contained in the database files, *.UDB files, that you extracted from each of the systems.



Java Package Directory: C:\Program Files\Eurekify\Eurekify Sage Client Tools V3.2\Software\UUID

UUID Mapping File: C:\test\uuid demo\SimpleSettings.xml

UUID Working Directories
Data Directory: C:\test\uuid demo\demodata
Index Directory: C:\test\uuid demo\demoindex
Output Directory: C:\test\uuid demo\demooutput

User Databases

XML FIle	UDB FIle	Master
SimpleMaster.xml	UsersDB.udb	YES
SimpleRACF.xml	RACF.cfg	
SimpleWinNT.xml	WinNT.cfg	
SimpleSolaris.xml	Solaris.cfg	

Match Process
Output Config: c:\test\UUID Demo\Simplematchcfg.cfg
Score Resolution: 1 Max Matches Allowed: Mismatch Level Allowed: 1 Minimum Score:

Merge Process
Output Config: c:\test\UUID Demo\SimpleMergedcfg.cfg

Master vs. Other Databases

The Master database is usually the database that you extracted from the system that supports your Human Resources department. Using the User Databases window you create virtual connections between each User Database file and a Master Database file based on common information contained in the Master Database and any of the other databases.

Databases extracted from Human Resources generally contain a broad set of data on the personnel in your organization and generally reference each person by a unique employee ID. This ID is the single piece of information that must be included in a Master Database. In most cases, more information will allow you to match more accounts more accurately. Thus, any other information that is available is important to be included in the Master Database: name, department, title, location, manager, and so on.

Connecting Master and Other Databases

To correlate between users in different databases, definitions are required that describe and "canonize" the user-related information contained in the databases. Those definitions are called UUID-Fields. Specifically, the NAME, GROUP and FUNCTION attributes of the UUID-Fields defined for each database provide a means to correlate the data.

Using these UUID-Field attributes you create a virtual bridge between each User Database and the Master Database. When the UUID tool processes the data in each of the databases, it uses the information in these virtual bridges to identify each person in the organization with the accounts on each system to which they have access.

In practice the virtual bridge is referred to as the Group attribute of the UUID-Field, and the Name and Function attributes define the actions that are performed on each field in the databases to correlate data between the Master Database and the other User Databases. To successfully match organization personnel with their accounts, you must examine each of the User Databases and create as many UUID-Fields as are needed to link each person listed in the Master Database to the accounts that are referenced in the User Databases.

Example Database Usage and UUID-Field Construction

This example shows two separate databases that treat data for a single employee in an organization. In Database 1 the employee is referenced by Person Name and the employee Telephone number is provided in the form <Area Code-Number>. In Database 2 the employee is referenced by a Person ID and the employee phone number is provided as two separate fields, Area Code and Phone Number.

Database 1

Fields in Database 1	Person Name	Telephone
Data	John Smith	09-7693219

Database 2

Fields in Database 2	Person ID	Area Code	Phone Number
Data	1234567	09	7693219

By looking at the phone number in each database you can see that the Phone Numbers are identical even though they are referred to in slightly different forms. We can therefore extrapolate from that, that the employee John Smith in Database 1 is the same individual that is referred to by the Person ID of 1234567 in Database 2. Essentially we have used the data provided by the phone numbers to build a virtual bridge between the two databases.

UUID-Field Construction in the UUID Tool

In the UUID tool, the Group attribute of the UUID-Fields forms the virtual bridge. You create UUID-Fields with given Group attributes in the Master Database for each type of information that you want to use. You then create UUID-Fields with identical Group attributes in the User Databases that contain the same type of information that you want to relate to the information in the Master Database. The functions may vary in structure for the identical Groups in each database, but the goal is to construct the same data set using the available fields in the databases. In our simple example, the databases looks as follows:

Database 1 UUID-Fields

Name	Group	Function
Database 1_Ex	Phone	Telephone

Database 2 UUID-Fields

Name	Group	Function
Database 2_Ex	Phone	<Area Code>-<Phone Number>

Each database contains a UUID-Field with a Group called Phone. The Functions for each Group vary in structure but the outcome is identical. In the case of the example a phone number that is in the form <Area Code>-<Phone Number>.

UUID-Field Elements

Each database can contain several UUID-Fields. Each UUID-Fields has the following elements: Name, Group, Function, and Weight. The following list describes these elements:

Name

Specifies a name that is provided for each UUID-Field that is extracted from the database. The name does not have to be identical across each database.

Group

Specifies a name that is used for each common data type. The name for each common data type must be identical in each database.

Function

Specifies the action to be performed on the database fields. This might be to extract the data contained in a database field, or it might be to extract a combination of data contained in several fields in the database.

For help on the protocol used to construct combinations click the ? button in the Fields section of the User Database window. Refer to ???? for a complete list of the functions available to manipulate database fields and create UUID-Fields.

Weight

Provides a numeric measure to indicate the internal priority given to each group within a database. The greater the value the higher the priority. The UUID tool processes the groups according to their order of priority.

A value of 0 means that this group is not taken into consideration in the matching process.

Naming UUID-Fields

Each database must contain at least one UUID-Field that references the field in the database that contains the user-account information (Login). The name provided for that UUID-Field must be provided in the following form: <Database Name>_ID. The Name provided for any other UUID-Field can take any form.

For example, for a database called RACF.udb the Name provided for the UUID-Field relating to the user-account field is RACF_ID.

The purpose of this special UUID-Field is to support the Merge operation (post matching). It is used to compare to the Person ID field in the merged configuration.

Note: The ID UUID-Field *is not* used for the correlation process. It should be associated with a group of its own, and given a weight of 0.

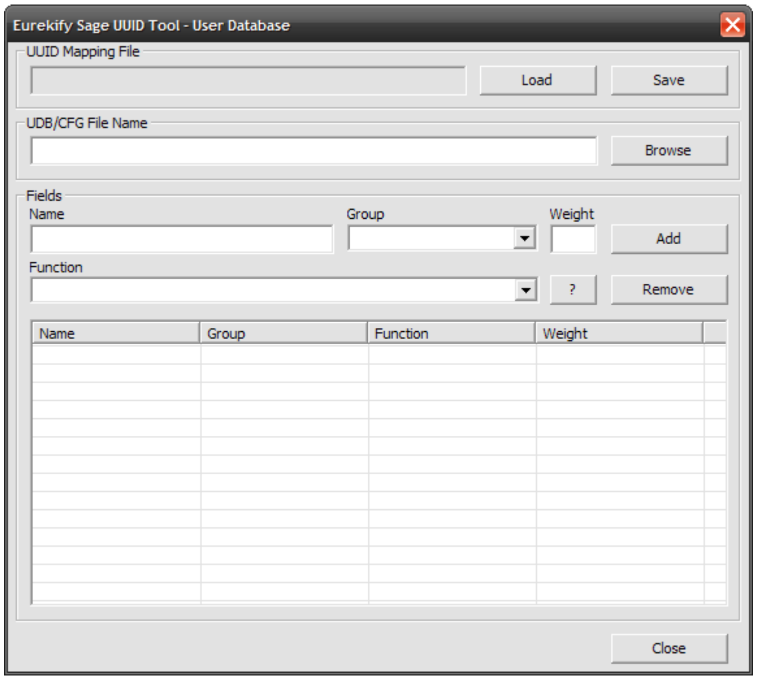
Adding New Databases

You need to include a database for each system in your organization that you are referencing. These are files that were extracted from each system and renamed as *.UDB files.

To add a new database

1. Click Add New in the User Databases section of the UUID Tool.

   The User Database window opens.



2. Click Browse next to the UDB/CFG File Name text field and from the Open dialog box select the database file that you want to include.

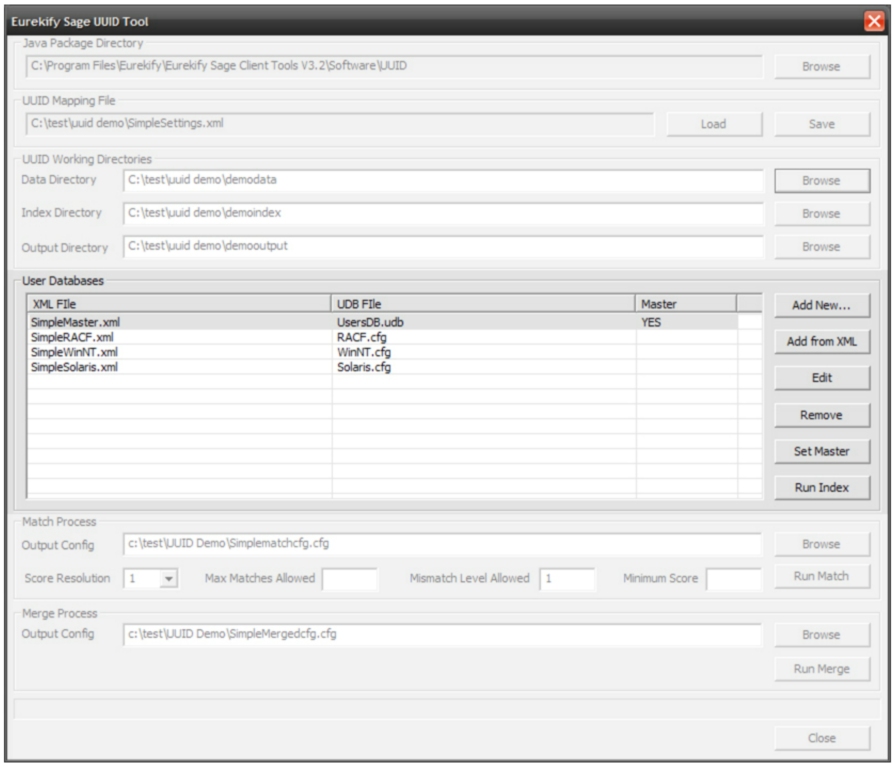
   Note: If you later plan to run the Merge Process, you need to select a Eurekify configuration file (.cfg file) originating from the referenced systems. Configuration files automatically direct the tool to their User Database (.udb file). Otherwise, you can select the User Database (.udb file) directly.

3. Click Open and the selected file name is displayed in the UDB/CFG File Name text field.

4. Click Save and provide a name for an XML file in the Save As dialog box. The XML file is the UUID Mapping file and stores all the mapping parameters associated with the database.

5. Repeat this procedure to add a reference for each User Database that was extracted from the organization.

   The following screen shows references to User Databases for each system treated in an organization, these include: UsersDB, RACF, WinNT and Solaris.



Java Package Directory: C:\Program Files\Eurekify\Eurekify Sage Client Tools V3.2\Software\UUID

UUID Mapping File: C:\test\uuid demo\SimpleSettings.xml

UUID Working Directories
Data Directory: C:\test\uuid demo\demodata
Index Directory: C:\test\uuid demo\demoindex
Output Directory: C:\test\uuid demo\demooutput

User Databases

XML FIle	UDB FIle	Master
SimpleMaster.xml	UsersDB.udb	YES
SimpleRACF.xml	RACF.cfg	
SimpleWinNT.xml	WinNT.cfg	
SimpleSolaris.xml	Solaris.cfg	

Match Process
Output Config: c:\test\UUID Demo\Simplematchcfg.cfg
Score Resolution: 1 Max Matches Allowed: Mismatch Level Allowed: 1 Minimum Score:

Merge Process
Output Config: c:\test\UUID Demo\SimpleMergedcfg.cfg

6. Select the Database that contains the HR data and click Set Master. This sets the selected database as the Master database.

The database that you select as the Master database must contain an explicit reference to each of your personnel by name. For this reason it is usually the database that contains the HR data.

Adding Databases from XML Files

If you already have an XML file from a previous implementation, you can refer to that XML directly. You do so by using the Add from XML feature in the User Databases section of the UUID tool.

To add a database from an XML file

1. From the User Databases section click Add from XML.

   The Save As window opens.

2. Navigate to the folder that contains your databases saved as XML files and select the database to add to the mapping file.
3. Click Save.

   The database is added to the list of User Databases referenced in the mapping file.

4. Click Save in the UUID Mapping File section to save the modified list of databases as part of the mapping file.

Editing Database UUID-Fields

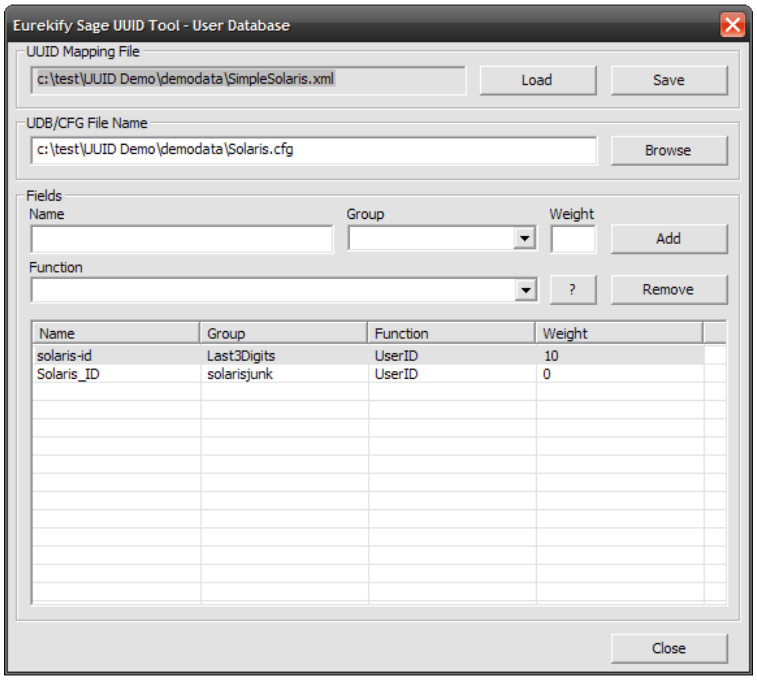
At times you may need to modify existing matching UUID-Fields in a database, add UUID-Fields to a database, or remove UUID-Fields from a database. You do so by using the Edit feature in the User Databases section of the UUID tool.

To edit a database UUID-Field

1. Select an XML file from the User Databases list.
2. From the User Databases section click Edit.

   The User Database window opens displaying the list of UUID-Fields.
3. Select the UUID-Field that you want to edit.

   The selected row is highlighted.



Eurekify Sage UUID Tool - User Database

UUID Mapping File: c:\test\UUID Demo\demodata\SimpleSolaris.xml

UDB/CFG File Name: c:\test\UUID Demo\demodata\Solaris.cfg

Name	Group	Function	Weight
solaris-id	Last3Digits	UserID	10
Solaris_ID	solarisjunk	UserID	0

4. Double-click in any field and the field becomes editable. You now can manually edit the value for the selected field.
5. When you are satisfied with your changes, click Save to confirm your changes in the database.

To add a UUID-Field to a database

1. Select an XML file from the User Databases list.
2. From the User Databases section click Edit.

   The User Database window opens displaying the list of UUID-Fields.
3. Enter values in the Name, Group and Function fields.
4. Enter a numeric value in the Weight text field.
5. Click Add.

   The new UUID-Field is added to the list of groups in the database.
6. Click Save to confirm your changes in the database.

To remove a UUID-Field from a database

1. Select an XML file from the User Databases list.
2. From the User Databases section click Edit.

   The User Database window opens displaying the list of groups.
3. Select the UUID-Field that you want to remove.

   The selected row is highlighted.
4. Click Remove and the selected group is deleted from the list of groups.
5. Click Save to confirm your changes in the database.

Note: You can define several UUID-Fields having the same Group name. For example, if the Master Database contains a value for US State (such as, NY), but it does not exist in a given User Database, you can still use some of the information that is available in the User Database to match to it. For example, suppose that the User Database contains telephone number and zip code. In that case, you can create two fields in the User Database: one will try to "guess" the state by mapping (lookup function) the telephone area code, and one will do the same but with the zip. Hopefully at least one of the matches will succeed and you will get a match.

Removing Databases

For any number of reasons you may no longer need to deal with data that is included in a particular system in your organization. In such cases you need to remove references in your mapping file to the database. You do so by using the Remove feature in the User Databases section of the UUID tool.

To remove a database from a mapping file

1. In the UUID tool, load the mapping file that contains the databases to be removed.

   The User Databases referenced in the mapping file are displayed in the User Databases list.

2. Select the User Database to be removed from the mapping file.

   The selected row is highlighted.

3. Click Remove.

   The selected row is deleted from the list.

4. In the UUID Mapping File section click Save to confirm the changes made to the mapping file.

Indexing the Databases

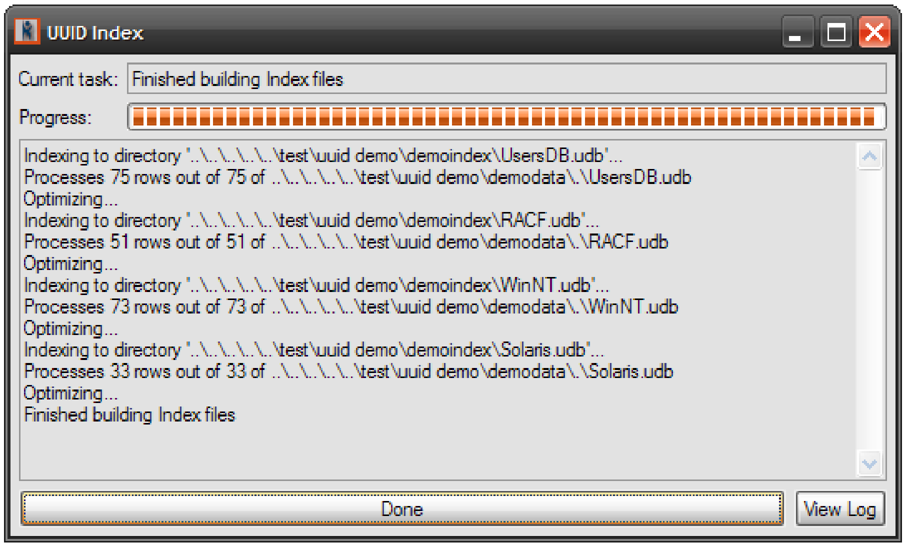
Index the databases referenced in a Mapping file you run the Match or Merge processes. While indexing the databases the UUID tool scans the data in each of the databases and loads the data into temporary files that are recorded in the Index Directory. If any changes are made to the database files or the Mapping file, then perform the index process again before you perform the Match or Merge process.

To index the databases

1. After setting the Working Directories, and defining the User Databases in the UUID tool, save the definitions as a Mapping file. If a Mapping file already exists click Load and load the mapping file into the UUID tool.
2. In the User Databases section of the UUID tool click Run Index.

   The UUID Index window opens and displays a progress bar for the index process. Depending on the size of your databases this process may take a couple of minutes.

   If an error occurs during the index process, an error message is issued as part of the progress report displayed in the lower part of the UUID Index window, and the cause of the error is indicated in the log file.



Current task: Finished building Index files

```
Indexing to directory '..\..\..\..\..\test\uuid demo\demoindex\UsersDB.udb'...
Processes 75 rows out of 75 of ..\..\..\..\..\test\uuid demo\demodata\.\UsersDB.udb
Optimizing...
Indexing to directory '..\..\..\..\..\test\uuid demo\demoindex\RACF.udb'...
Processes 51 rows out of 51 of ..\..\..\..\..\test\uuid demo\demodata\.\RACF.udb
Optimizing...
Indexing to directory '..\..\..\..\..\test\uuid demo\demoindex\WinNT.udb'...
Processes 73 rows out of 73 of ..\..\..\..\..\test\uuid demo\demodata\.\WinNT.udb
Optimizing...
Indexing to directory '..\..\..\..\..\test\uuid demo\demoindex\Solaris.udb'...
Processes 33 rows out of 33 of ..\..\..\..\..\test\uuid demo\demodata\.\Solaris.udb
Optimizing...
Finished building Index files
```

If you neglected to Save the mapping file prior to trying to Run Index, a Save As window opens for you to save the file. After saving the file the UUID Index process begins automatically.

3. (Optional) To view a log of the index process click View Log to open the log. The log contains a line for each record that was scanned in each of the databases included in the mapping file.

   At the end of the progress display, the message Finished building Index files is displayed when the index is successfully built.

4. Click Done when the Index process is complete.

UUID Mapping File

The UUID Mapping File is an XML file that stores the parameters that are set in the UUID Working Directories, User Database, Match Process and Merge Process sections of the UUID tool. Once the parameters are saved, you can use the Mapping file to quickly populate the UUID Tool with the saved parameters instead of manually entering the data each time that you want to run the Match or Merge process. Alternately you can load mapping file and use it as the base for editing and saving a new mapping file under a new name.

To use a UUID Mapping File

1. Click Load in the UUID Mapping File section of the UUID tool.

   An Open dialog appears in which you can navigate to the location that contains the mapping files on your local machine. For organizational purposes we suggest that the UUID Mapping Files be saved in the same directory that contains the Working Directories.

2. Select an XML and click Open.

   The parameters stored in the XML file are loaded into the UUID tool.

Match Process

The Match process reads the User Database files referenced in the User Databases section of the UUID tool and correlates the Users with the account details in each of the systems. The results of the Match Process are stored in a configuration file.

To run the Match Process

1. Click Load in the UUID Mapping file section and load a Mapping XML file.

   The UUID Tool is populated with the parameters stored in the selected UUID file.

2. Click Run Index in the Users Databases section.

   The listed User Databases are indexed. Depending on the size of the Databases the indexing process may take a few minutes.

3. Click Run Match in the Match Process section.

The UUID tool processes the databases and tries to correlate every account in each User Database to one or more potential owners in the Master Database. The correlation is based on the fields defined for matching, weighted accordingly. The result is a Matching Configuration, where each of the users in the Master Database appears In the configuration's User Pane, and each of the users in the other User Databases (representing accounts) appear in the configuration's Resource. Res Name 1 is the account ID, taken from the <Database Name>_ID field in the User Database The name of the source system appears as Res Name 2. The degree of match is represented in the score (0-100) and appears as Res Name 3.This information is saved in the configuration file listed in the Output Config field of the Match Process section. You can now open the Output Configuration file in Sage DNA and view each person in the organization and the accounts on each system to which they have access.

Because the matches are represented as a regular Eurekify configuration, you can also:

- Review and add/remove/change correlations manually, using the Sage DNA Workstation
- Report all correlations, using the Eurekify Reporting facilities
- Run a certification campaign to confirm the correlations, using the Eurekify Portal

See the respective user manuals for more details.

When reviewing and correcting correlation in the Sage DNA Workstation, pay special attention to:

- Accounts that were not matched at all (Res Name 3 will be empty for these)
- Accounts that were matched but with a low probability (low score in Res Name 3) and thus represent more of a guess than a deterministic matching
- Accounts that were matched to multiple people (first note accounts with Total Number of Users greater than 1; note also that same account may be matched with different scores, so look out for those as well).

Merge Process

After you run the Match Process, inspect the results, and perform needed corrections, you now have a finalized configuration file, matching each person in the organization with their respective accounts on the referenced systems. You can now proceed to the final stage of creating a final configuration that links each person in the organization with all their resources in the referenced systems. This phase is called the Merge Process.

The Merge process reads the configuration files referenced in the User Databases section of the UUID tool and correlates the Users with the resource details in each of the systems that are referred to in the tool.

Note: To run the Merge Process, the UUID tool needs to have access to the configuration files of the referenced systems (.cfg files), and not to the Users Databases (.udb files).

To run the Merge Process

1. Click Load in the UUID Mapping file section and load a Mapping XML file.

   The UUID Tool is populated with the parameters stored in the selected UUID file.

   We assume that you have previously run a Match Process and that the configuration specified in the Output Config field of the Match Process section exists and represents the correct matching.

2. Click Run Merge in the Merge Process section.

   Eurekify UUID processes the databases and matches each person in the organization with the resources to which they have access rights and privileges across each system in the organization. This information is saved in the configuration file listed in the Output Config field of the Merge Section.

3. You can now open the Output Configuration file in Sage DNA and view the each person in the organization and the resources on each system to which they have access.

Chapter 2: UUID Indexing Functions

This section contains the following topics:

UDB Fields Referencing

UDB fields can be referenced directly, for example *FirstName*, or with the Field Function, such as Field*('FirstName')*.

If the UDB field contains a space (' ') character, it can only be referenced with the FIELD function. for example *Field('User Name')*.

Field Referencing

Function name	**Parameters**
Example	**Results**
<Direct>	Param1 - field name
FirstName	'John'
Field(fieldname)	fieldname -name of a field from the UDB
Field('First Name')	'John'

Lookup Functions

Translating using a CSV file

Function name	**Parameters**
Example	**Results**
CsvLookup(csvFilename, value)	csvFilename - the CSV file containing the translation map value - the value to look-up

CsvLookup('areas.csv', City)	

String Functions

String Concatenation

Function name	**Parameters**
Example	**Results**
+ operator	str1 - string str2 - string
FirstName + LastName	'John Smith'

String Concatenation

Function name	**Parameters**
Example	**Results**
Concat(str1, str2, separator)	str1 - string str2 - string separator - string
Concat('Hello','world',', ')	'Hello, world'

Sub String

Function name	**Parameters**
Example	**Results**
Substr(str,from,to)	str - the string from - starting offset of requested substring to - ending offset of requested substring
Substr('John Smith',5,6)	'Sm'

String Trimming

Function name	**Parameters**
Example	**Results**
Trim(str)	str - string with leading/ending spaces
Trim(' sentence between many spaces ')	'sentence between many spaces'

String Last Characters

Function name	**Parameters**
Example	**Results**
LastChars(str,len)	str - string len - integer value specifying the required length of the tail
LastChars('where is the end',7)	'the end'

String Length

Function name	**Parameters**
Example	**Results**
Strlen(str)	str - string
Strlen('hello world')	11

String Searching

Function name	**Parameters**
Example	**Results**
StrFind(str,substr)	str - string substr - the string which we need offset of
StrFind('My favorite color is red','color')	12

Convert from Integer to String

Function name	**Parameters**
Example	**Results**
StrOf(int)	int - integer value
StrOf(5)	'5'

Finding Digits in a String

Function name	**Parameters**
Example	**Results**
DigitsOf(str)	str - string
DigitsOf('john12smith34')	'1234'

Replacing Strings

Function name	Parameters
Example	**Results**
StrReplace(strSource,substr,replacing)	strSource - source string substr - the substring to be replaced replacing - the new sub-string
StrReplace('firstname1lastname1','1','2')	'firstname2lastname2'

Finding Alphabetic Characters

Function name	Parameters
Example	**Results**
AlphaOf(str)	str - string
AlphaOf('a1!@b2#$A1%^B2')	'abAB'

Finding Alpha-Numeric Characters

Function name	Parameters
Example	**Results**
AlphaAndDigitsOf(str)	str - string
AlphaAndDigitsOf('a1!@b2#$A1%^B2')	'a1b2A1B2'

Lower Case Conversion

Function name	Parameters
Example	**Results**
ToLower(str)	str - string
ToLower('RRYMON')	'rrymon'

Upper Case Conversion

Function name	Parameters
Example	Results
ToUpper(str)	str - string
ToUpper('rrymon')	'RRYMON'

Two-way Case Conversion

Function name	Parameters
Example	**Results**
SwapCases(str)	str - string
SwapCases('RRymon')	'rrYMON'

Removing Vowels from a String

Function name	Parameters
Example	**Results**
RemoveVowels(str)	str - string
RemoveVowels('johnSMITH')	'jhnSMTH'

Left-to-Right Reversing

Function name	Parameters
Example	**Results**
Reverse(str)	str - string
Reverse('john SMITH')	'HTIMS nhoj'

Telephone Number Functions

Finding Country Code

Function name	Parameters
Example	**Results**
TelCountryCode(phone)	phone - full phone number
TelCountryCode('+972-8-7654321')	'972'

Finding Area Code

Function name	Parameters
Example	**Results**
TelAreaCode(phone)	phone - full phone number
TelAreaCode('+972-8-7654321')	'8'

Finding last 7 Digits of a Phone Number

Function name	Parameters
Example	**Results**
Tel7Digits(phone)	phone - full phone number

Tel7Digits('+972-9-7467346')	'7467346'

Name Functions

Getting First Name

Function name	Parameters
Example	**Results**
FirstName(name)	name - full name
FirstName('Ron Rymon')	'Ron'

Getting Last Name

Function name	Parameters
Example	**Results**
LastName(name)	Name - full name
LastName('Ron Rymon')	'Rymon'

Getting Middle Name

Function name	Parameters
Example	**Results**
MiddleName(name)	Name - full name
MiddleName('Ron Rymon')	'' (empty string)
MiddleName('John Ferdinand Smith')	'Ferdinand'

Getting Middle Initial

Function name	Parameters
Example	**Results**
MiddleInitial(name)	Name - full name
MiddleInitial('John Ferdinand Smith')	'F'

Getting Name Suffix

Function name	Parameters
Example	**Results**
NameSuffix(name)	Name - full name, including suffix
NameSuffix ('John Smith, Jr.')	'Jr.'

Email Address Functions

Getting User ID from Email Address

Function name	Parameters
Example	**Results**
EmailUserID(emailAddress)	emailAddress - full email address
EmailUserID('rrymon@eurekify.com')	'rrymon'

Getting Email Domain From Email Address

Function name	Parameters
Example	**Results**
EmailDomain(emailAddress)	emailAddress - full email address
EmailDomain('rrymon@eurekify.com')	'eurekify.com'

Formatting Email Address

Function name	Parameters
Example	**Results**
EmailConvention (format, first, last, domain)	Create a convention formatted string of email address format - one of: ■ Flast ■ Lastf ■ First.last ■ Last.first ■ Last ■ First first - first name last - last name domain - the email domain
EmailConvention('flast','John', 'Smith', 'eurekify.com')	'jsmith@eurekify.com'
EmailConvention('lastf','John', 'Smith', 'eurekify.com')	'smithj@eurekify.com'
EmailConvention('first.last','John', 'Smith', 'eurekify.com')	'john.smith@eurekify.com'

EmailConvention('first_last','John', 'Smith', 'eurekify.com')	'john_smith@eurekify.com'
EmailConvention('last','John', 'Smith', 'eurekify.com')	'smith@eurekify.com'
EmailConvention('first','John', 'Smith', 'eurekify.com')	'john@eurekify.com'

Address Functions

Getting Country Name from Address

Function name	Parameters
Example	**Results**
AddressCountry(fullAddress)	fullAddress - string of full address
AddressCountry('Eurekify Ltd. Hasadna 82, Floor 1, Raanana, ISRAEL 46345')	

Getting City From Address

Function name	Parameters
Example	**Results**
AddressCity(fullAddress)	fullAddress - string of full address
AddressCity('Eurekify Ltd. Hasadna 82, Floor 1, Raanana, ISRAEL 46345')	

Getting Street Name from Address

Function name	Parameters
Example	**Results**
AddressStreet(fullAddress)	fullAddress - string of full address
AddressStreet('Eurekify Ltd. Hasadna 82, Floor 1, Raanana, ISRAEL 46345')	

Getting State from Address

Function name	Parameters
Example	**Results**
AddressState(fullAddress)	fullAddress - string of full address
AddressState('Eurekify Ltd. Hasadna 82, Floor 1, Raanana, ISRAEL 46345')	

String

Function name	Parameters
Example	**Results**
AddressZipCode(fullAddress)	fullAddress - string of full address
AddressZipCode('Eurekify Ltd. Hasadna 82, Floor 1, Raanana, ISRAEL 46345')	'46345'

Getting All Digits from an Address

Function name	Parameters
Example	**Results**
AddressDigits(fullAddress)	fullAddress - string of full address
AddressDigits('Eurekify Ltd. Hasadna 82, Floor 1, Raanana, ISRAEL 46345')	'82 1 46345'

Function Composition

It is possible to compose functions, for example: ToLower(AlphaOf('A1B2C3')) => 'abc'

User-Defined Functions

It is possible for users to use their own defined and implemented functions.

The declaration of such functions is done in an XML file named "userJarsDef.xml". The format of the file is:

```
[set the jars variable for your book]
  <indexFunction
jarFilename="c:\dev\uuid\userJar.jar"
implClass="EvalSubstring"
function="UserPrivateSubstring" />
</jars>
```

The function implementation is expected to be found in the specified JAR file. The specified class should extend the class:

com.eurekify.matcher.indexer.evalfunctions. EvalFunc

The default-constructor of the class should define the number parameters this function accepts:

```
   numberOfParameters = 1;
```

The class should implement the method:

```
public void run(Stack<Object> stack) throws EurekifyEvaluationException
```

First the stack needs to be checked with the function

```
   checkTheStack(stack);
```

The parameters passed to the function are retrieved from the stack using:

```
   String strParam = getStringParam(stack);
```

Or:

```
   int intParam = getIntParam(stack);
```

The result of the function should be pushed back to the stack using

```
   stack.push(result);
```

Example implementation class:

```
import java.util.Stack;
import com.eurekify.matcher.indexer.evalfunctions.*;
public class EvalSubstring extends EvalFunc {
  public EvalSubstring() {
    numberOfParameters = 3;
  }
  public void run(Stack<Object> inStack) throws EurekifyEvaluationException
  {
  // check the stack
  checkTheStack(inStack);
  // get the parameter from the stack
  int to = getIntParam(inStack);
  int from = getIntParam(inStack);
  String str = getStringParam(inStack);

  String result = str.substring(from, to);
  // push the result on the stack
  inStack.push(result);
   }
}
```