Directory Sync

Advanced Mappings Guide

Legend

! **CAUTION: A CAUTION icon indicates potential damage to hardware or loss of data if instructions are not followed.**

i **IMPORTANT**, **NOTE**, **TIP**, **MOBILE**, or **VIDEO:** An information icon indicates supporting information.

Directory Sync – Advanced Mappings Guide
Updated – March 2024

Contents

Template Advanced Mappings **5**

What is a Value? **5**

What is a Condition? **6**

Text strings and special characters **6**

Data sources **6**
- **Source** 7
- **Target** 7
- **Settings in Templates or Stage Data Steps (also referred to as a Profile)** 7

Operators **7**
- **Math** 7
- **Value Comparison** 7
- **Boolean Combinations** 8
- **Boolean Constants** 8
- **case** 8
- **if** 8

Text Operations **9**
- **empty** 9
- **contains** 9
- **ends** 9
- **length** 10
- **lower** 10
- **replace** 10
- **starts** 10
- **trim** 11
- **trunc** 11
- **upper** 11
- **index** 11
- **Prefix** 12
- **Suffix** 12
- **Right** 12
- **Left** 12
- **Substring** 12
- **GetList** 13
- **UpdateList** 13
- **ToString** 13

Template and Stage Data Step Settings **14**
- **Profile.TargetDomain** 14
- **Profile.DefaultPassword** 14
- **AllowTargetAddress** 15
- **GetGroupType** 15
- **GetDN** 15
- **GetUserAccountControl** 16

General Functions **17**
- **Lookupvalue** 17
- **encodePWD** 17
- **GetAccountDisabled** 17
- **ReplaceDomain** 17
- **GetProxyAddresses** 18
- **GetLegacyExchangeDN** 18
- **GetObjectClass** 19
- **GetTargetAddress** 19
- **GetRecipientTypeDetails** 19
- **Action** 19
- **result** 20
- **Target.HasCreateStamp** 20
- **ConvertValue("base64STR", objectGUID)** 20
- **NewGUID()** 20

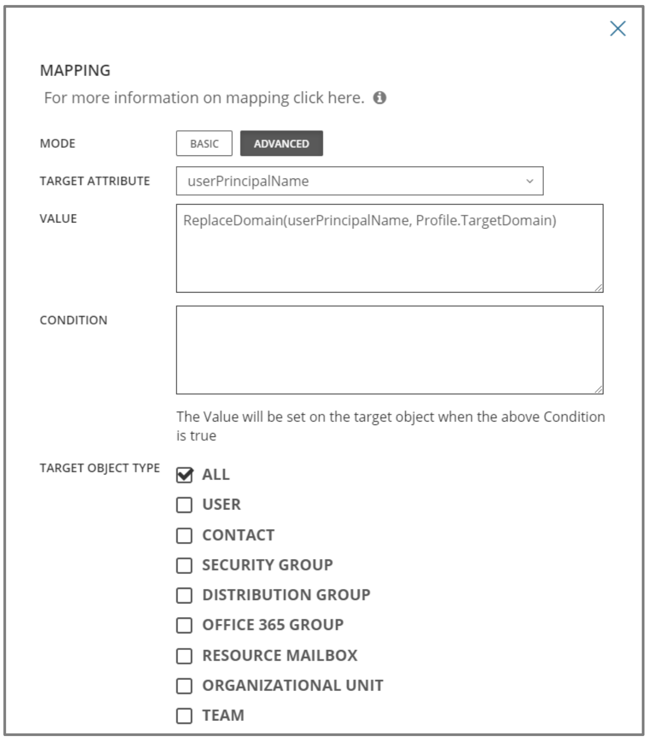
Default Advanced Mappings **21**
- **Local to Local** 21
- **Cloud to Local** 22
- **Cloud to Cloud** 23
- **Local to Cloud** 24

About us **26**

1

Template Advanced Mappings

In addition to basic mappings, you can perform advanced mappings for target attributes using values and conditions within a template's Mapping settings. Advanced mappings provide the flexibility needed to meet complex scenarios.



MAPPING

For more information on mapping click here.

MODE: BASIC | ADVANCED

TARGET ATTRIBUTE: userPrincipalName

VALUE: ReplaceDomain(userPrincipalName, Profile.TargetDomain)

CONDITION:

The Value will be set on the target object when the above Condition is true

TARGET OBJECT TYPE

- [x] ALL
- [ ] USER
- [ ] CONTACT
- [ ] SECURITY GROUP
- [ ] DISTRIBUTION GROUP
- [ ] OFFICE 365 GROUP
- [ ] RESOURCE MAILBOX
- [ ] ORGANIZATIONAL UNIT
- [ ] TEAM

What is a Value?

A value is a formula or single value which determines the value that is set on the target attribute of the mapping. A combination of functions, text, and/or operations can be used to construct a Value. Values are limited to 2000 characters in length.

Other important items to note:

- A target attribute can only exist once in the mapping table. When creating values ensure that you can cover the requirements for that attribute with one value.

- Some default values exist and are detailed later in this document. It is important that you have a complete understanding of these default values before making any changes to them. It is suggested to not make changes to the default values if you do not have experience with advanced mappings.

What is a Condition?

A Condition works like an on/off switch. If the formula entered in the Condition box evaluates to true or the box is empty, the value in the Value box will be applied to the target attribute. If in any case the formula in the Condition box is false, the value will not be applied. An example of using a condition would be setting the Condition field to Action = "Create" when you want to set a Value only when creating a user.

Text strings and special characters

When using Attributes with names containing special characters (matching any operator) they must be enclosed in square brackets ([and]), otherwise they will be interpreted as the corresponding operator, such as a minus sign rather than a dash.

Example:

[msDS-PhoneticFirstName]

Text strings must be enclosed in double quotation marks e.g. ("sample").

All values of Attributes and text strings are case sensitive.

Examples:

Evaluating "Test user" = "test user" will result in false.

Evaluating "Test user" = "Test user" will result in true.

Data sources

Within the advanced mappings you can use attributes from the source or target environments. You can also use the settings defined within the stage data step (also referred to as a profile). The current action being performed on the object can also be used.

By **default,** if the name of an attribute is entered with no prefix then it is assumed that you would like to use the value of that attribute from the source environment.

To specify source or target environment the following prefix methods can be used

Source

- Source or S followed by a period in front of the attribute name returns the value of source object.

  Examples: Source.DisplayName, S.DisplayName, or DisplayName

Target

- Target or T followed by a period in front of the attribute name returns the value of target object.

  Examples: Target.DisplayName, T.DisplayName

Settings in Templates or Stage Data Steps (also referred to as a Profile)

- Profile or P followed by a period in front of a setting name returns the value of a setting in templates or stage data steps.

  Example: Profile.DefaultPassword returns the default password value from the settings.

Operators

Operators are used to manipulate or compare values. You can do basic Math, Value Comparisons, and conditional flow control with "if" and "case" statements. You can combine operations and compare with Boolean operators.

Math

- Purpose: The following can be used to perform calculations of numeric values or combine string-based values.
- Syntax:

  + : Add numbers or concatenate strings

  − : Subtract numbers

  * : Multiply numbers

  ⁄ : Divide numbers
- Note: Spaces are needed between symbols and values.
- Example 1: S.FirstName + S.SN
- Example 2: S.GivenName + "." + S.LastName
- Example 3: 12 = 5 + 7

Value Comparison

- Purpose: The following operators return True or False when comparing values.
- Syntax:

  = : Are two values are equal?

> : Is the left value greater than the right value?

>= : Is the left value greater than or equal to the right value?

< : Is the left value is less than the right value?

<= : Is the left value less than or equal to the right value?

!= : Is the left value not equal to the right value?

- Note: Spaces are needed between symbols and values.
- Example : if(length(S.GivenName) > 10, S.GivenName, S.sn)

Boolean Combinations

- Purpose: Combinations of operations can be achieved with the following.
- Syntax:

  And

  Or

  ! (not)

- Example: Action = "create" or (Action = "update" and Target.HasCreateStamp)

Boolean Constants

- Purpose: Returns pure values for Boolean logic.
- Syntax:

  True

  False

  Null (empty)

- Example: You want to set the target DisplayName to the source value only if it is Null
- if(T.DisplayName = Null, S.DisplayName, T.DisplayName)

case

- Purpose: Branching logic flow statement, evaluates different statements based on a given conditions, evaluated in order.
- Syntax: case(condition, value, condition, value, ...)
- Example: You want to set the department attribute to a standard, but the source value does not match your new standard. Departments in the source may be "IT" and "FIN", but in the target they need to be "Info Tech" and "Finance". Use of the case statement below will set department to "Info Tech" when the source department equals "IT" and set the department to "Finance" when the source department equals "FIN".

Target Attribute	: Department
Value	: case(S.Department = "IT","Info Tech", S.Department = "FIN","Finance")
Condition	: Null

if

- Purpose: Returns a specified value if given condition is true, otherwise returns a different specified value.
- Syntax: if(condition, value1, value2)

  returns value1 when condition evaluates to True, otherwise returns value2

- Example: You need to set your display name to include (MGR), but only if the title of the user contains (Manager). For non-managers, display name is simply first name and last name. The value expression below uses an If statement to check if the source attribute title contains the word "Manager". If it contains "Manager", it will append " (MGR)" to the end of the first and last names.

  Target Attribute : DisplayName

  Value : if(contains(s.title, "Manager"), s.firstName+s.lastName + " (MGR)", s.firstName + s.lastName)

  Condition : Null

Text Operations

Text Operations are functions used to manipulate or build text values. In some cases, such as the contains() and empty() operations, the text operation can be used to check for the presence of a given value.

empty

- Purpose: This method checks if a value is empty. Returns True if the given expression evaluates to either Null or an empty string ("")
- Syntax: empty(value)
- Example: The following will set the Company attribute to "BlueFish Hotels" if the Source company field is empty string or Null.

  Target Attribute : Company

  Value : "BlueFish Hotels"

  Condition : empty(S.Company)

contains

- Purpose: This method checks if the given string value exists anywhere within the given string source. Returns True if value is found in source and False otherwise.
- Syntax: contains(source, value)
- NOTE: Does not work with list or multivalued attributes.
- Example: You want to set ExtensionAttribute10 to "No Sync", but only if the source Company field contains "BlueFish".

  Target Attribute : ExtensionAttribute10

  Value : "No Sync"

  Condition : contains(S.Company, "BlueFish")

ends

- Purpose: This method evaluates whether the given string value is a suffix of the given string source. Returns True if source ends with value and False otherwise.
- Syntax: ends(source, value)
- Example: You are rebranding and need to change values in the target based on values in the source attribute Company. The following example will set the target Company attribute to "BlueFish Motels" if the source Company field ends with "Hotels"

Target Attribute : Company

Value : "BlueFish Motels"

Condition : ends(S.Company, "Hotels")

length

- Purpose: This method returns the length in characters of a given string value
- Syntax: length(value)
- Example: The standard for email alias is firstName.lastName unless that would exceed 10 characters. When it exceeds 10 characters only use the first initial and 8 characters from the last name. The following expression will check if the combination of source firstName and lastName is less than 10 characters in length. If so (length is less than 10) the value will be set to firstName.lastName. However, if the combination would contain more than 10 characters, the value will be set to firstInitial.lastName with the last name limited to the first 8 characters.

  Target Attribute : Alias

  Value : if(length(s.firstName + s.lastName) < 10, s.firstname+"."+ s.lastname, trunc(s.firstName, 1) + "." + trunc(s.lastName, 8))

  Condition : Null

lower

- Purpose: This method is used to convert all characters of a given string value to remove capitalization. This can be helpful prior to comparisons as well as in other situations which require lowercase wording. The return value is a new string containing the lowercase string.
- Syntax: lower(value)
- Example: The customer wants the target DisplayName to be all in lowercase.

  Target Attribute : DisplayName

  Value : lower(S.DisplayName)

  Condition : Null

replace

- Purpose: This method swaps all instances of one target string within a search string with a replacement string. Returns a new string created from given input string source with all instances of string value exchanged for instances of string replacement. If string source does not contain any instances of string value, the result will be the same as source. Note, the resulting string may be longer or shorter than the source string.
- Syntax: replace(source, value, replacement)
- Example: The following expression will replace part of the old company name with the new one. The goal is to replace "RedFish" in the source Company attribute with "BlueFish" in the target. If the source is "RedFishHotels" the new value in the target will be "BlueFishHotels".

  Target Attribute : Company

  Value : replace(S.Company, "RedFish", "BlueFish")

  Condition : Null

starts

- Purpose: This method evaluates whether the given string value is a prefix of the given string source. Returns True if source begins with value and False otherwise.
- Syntax: starts(source, value)
- Example: The following will set CustomAttribute10 to "NA" if the source Location attribute begins with "United States"

Target Attribute : CustomAttribute10

Value : "NA"

Condition : starts(S.Location, "United States")

trim

- Purpose: This method removes leading and trailing spaces from given string value. Note that this does not remove other whitespace characters besides space. The return value is a copy of the input string value with no leading or trailing spaces.
- Syntax: trim(value)
- Example: "BlueFishHotels" becomes "BlueFishHotels"

  Target Attribute : Company

  Value : trim(S.Company)

  Condition : Null

trunc

- Purpose: This method returns a string which contains only the given length number of characters starting from the beginning of the given string source. Note, if length is greater than the length of string source, the result will be identical to string source.
- Syntax: trunc(source, length)
- Example: You want to use firstName and lastName to build the sAMAccountName but are limited to 20 characters.

  Target Attribute : sAMAccountName

  Value : trunc(S.givenName+S.sn, 20)

  Condition : Null

upper

- Purpose: This method is used to convert the characters of a given string value to all capital letters. This can be helpful prior to comparisons as well as in other situations which require uppercase wording. The return value is a new string containing the uppercase string.
- Syntax: upper(value)
- Example: The customer wants the target DisplayName to be all in uppercase.

  Target Attribute : DisplayName

  Value : upper(S.DisplayName)

  Condition : Null

index

- Purpose: This method returns the location of the beginning of the first instance of given string value within given string source. The position begins counting from 1 for the first character. Returns -1 if no instance of string value is found in string source.
- Syntax: index(source, value)
- Example: You need to use the left part of the UPN as the DisplayName in a new environment. The following expression will find the location in the UPN string of the '@' and then take all characters to the left of that.

  Target Attribute : DisplayName

  Value : trunc(S.userPrincipalName, index(S.userPrincipalName, "@") - 1)

  Condition : Null

Prefix

- Purpose: The prefix function will add the prefix specified to the value specified ONLY if that value is NOT null or an empty string.
- Syntax: prefix(value,prefix)
- Example:

Target Attribute	: used in combination with other functions to create a proxyaddress
Value	: prefix(Result("mail"), "SMTP:")
Condition	: Null

Suffix

- Purpose: The Suffix function will add the suffix text specified to the value specified ONLY if that value is NOT null or an empty string. This can be used to append to a DisplayName
- Syntax: suffix(value, suffix)
- Example:

Target Attribute	: DisplayName
Value	: suffix(S.DisplayName, "(MGR)")
Condition	: Null

Right

- Purpose: The Right function will return characters from the end of the string.
- Syntax: right(Source, Count)
- Source - Value to operate on
- Count - Number of characters from the end of the string to return. If Count is longer than the string, the entire string is returned.
- Example: You want to return the last four characters from the source display name.

Target Attribute	: DisplayName
Value	: right(S.DisplayName, 4)
Condition	: Null

Left

- Purpose: The Left function will return characters from the beginning of the string.
- Syntax: left(Source, Count)
- Source - Value to operate on
- Count - Number of characters from the beginning of the string to return. If Count is longer than the string, the entire string is returned.
- Example: You want to return the first four characters from the source display name.

Target Attribute	: DisplayName
Value	: left(S.DisplayName, 4)
Condition	: Null

Substring

- Purpose: The Substring function will return a subset of characters from a string.
- Syntax: substring(Source, StartIndex, Length)

- Source - (Required) Value to operate on
- StartIndex - (Required) Zero based Start Index position (First character = 0)
- Length - (Optional) Number of characters from Start Index to return. If StartIndex + Length extends beyond the string, Length is ignored, and the rest of the string is returned instead.
- Example: You want to return the 4th through 9th characters from the source display name.

  Target Attribute : DisplayName

  Value : substring(S.DisplayName, 3, 5)

  Condition : Null

GetList

- Purpose: This method is used to create list of values to write to a multi-target attribute.
- Syntax: GetList(Value1, Value2,…)
- Example: The following expression will directly populate ShowInAddressBook values with custom address book lists instead of using the target environment’s Exchange Default address list.

  Value: GetList("CN=Default Global Address List,CN=All Global Address Lists,CN=Address Lists Container,CN=contoso,CN=Microsoft Exchange,CN=Services,CN=Configuration,DC=contoso,DC=com","CN=All Users,CN=All Address Lists,CN=Address Lists Container,CN=contoso,CN=Microsoft Exchange,CN=Services,CN=Configuration,DC=contoso,DC=com")

UpdateList

- Purpose: This method is used to update/replace an existing value in a multi-value attribute.
- Syntax: UpdateList(S.ListAttribute, "Value To Replace", "Replacement Value")
- Example 1: The following expression will replace the “Test1” value in the source msExchExtensionCustomAttribute1 attribute with “Test-Replaced” and set it in the target.

  Source “msExchExtensionCustomAttribute1” = Test1, Test2, Test3

  Target Attribute : msExchExtensionCustomAttribute1

  Value: UpdateList(S.msExchExtensionCustomAttribute1, “Test1”, “Test-Replaced”)
- Example 2: The following expression will replace the “Test1” value in the source msExchExtensionCustomAttribute1 attribute with “Test-Replaced”, Remove “Test2” and set it in the target.

  Source “msExchExtensionCustomAttribute1” = Test1, Test2, Test3

  Target Attribute : msExchExtensionCustomAttribute1

  Value: UpdateList(UpdateList(S.msExchExtensionCustomAttribute1, “Test1”, “Test-Replaced”),”Test2”,NULL)

ToString

- Purpose: This method is used to convert multi-value attribute values into a string separated by a delimiter and set it on a single value attribute
- Syntax: ToString(Delimiter, Value)
- Example: The following expression will convert source “msExchExtensionCustomAttribute1” attribute into a string, separated by “,” and set it on “CustomAttribute1” attribute

  Source “msExchExtensionCustomAttribute1” = Test1, Test2,Test3

  Target Attribute: customAttribute1

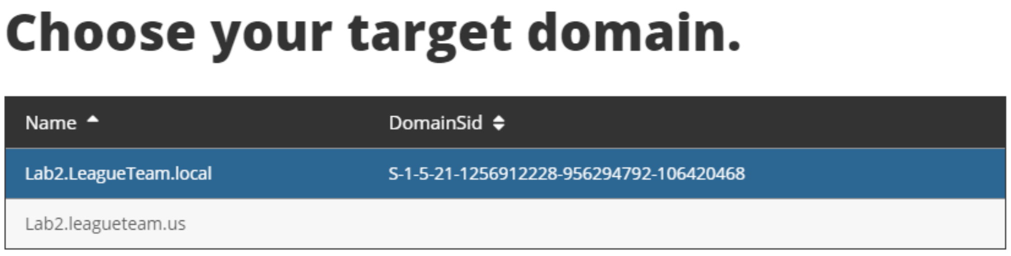
  Value: ToString(“,”,s. msExchExtensionCustomAttribute1)

Template and Stage Data Step Settings

These functions are used to gather or set values based on settings within the selected template and the Stage Data step of a workflow. Some functions will simply return a value and others will return a computed value based on which setting was selected. For example, Profile.TargetDomain returns a computed value which is a domain in the format of redfishhotels.com where getdn(CN) returns a complete value adding the source CN to the Target OU within the Stage Data setting. These functions all begin with uppercase letters to differentiate them from other functions.

Profile.TargetDomain

- Purpose: This profile setting will return the domain selected in the UI below during the configuration of the Stage Data step in a workflow. It will apply to the mapping template selected within the Stage Data Step.
- Syntax: Profile.TargetDomain or P.TargetDomain
- Location: The Stage Data Step of a workflow for Local and Cloud Environments


Choose your target domain.

Name	DomainSid
Lab2.LeagueTeam.local	S-1-5-21-1256912228-956294792-106420468
Lab2.leagueteam.us	

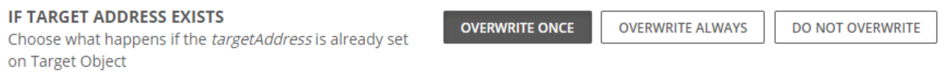
- Based on the above the use of Profile.TargetDomain or P.TargetDomain will result in a value of "TwoFishHotels.com"

Profile.DefaultPassword

- Purpose: This profile setting refers to the Default password set in the template used within a Stage Data step of a workflow and is used to encode a password so that it can be applied to newly created objects in AD
- Syntax: Profile.DefaultPassword or P.DefaultPassword
- Location: The Template Settings for Local and Cloud environments
- Based on the above the use of Profile.DefaultPassword or P.DefaultPassword would return the value entered and masked by the *'s. It will not be encoded, but rather be in plain text.

AllowTargetAddress

- Purpose: This function works like a profile setting where it returns the behavior defined within the template. Further details of this function can be found in the function section of this document.
- Syntax: AllowTargetAddress()
- Location: This profile setting refers to the setting selected by the user for targetAddress within the template.



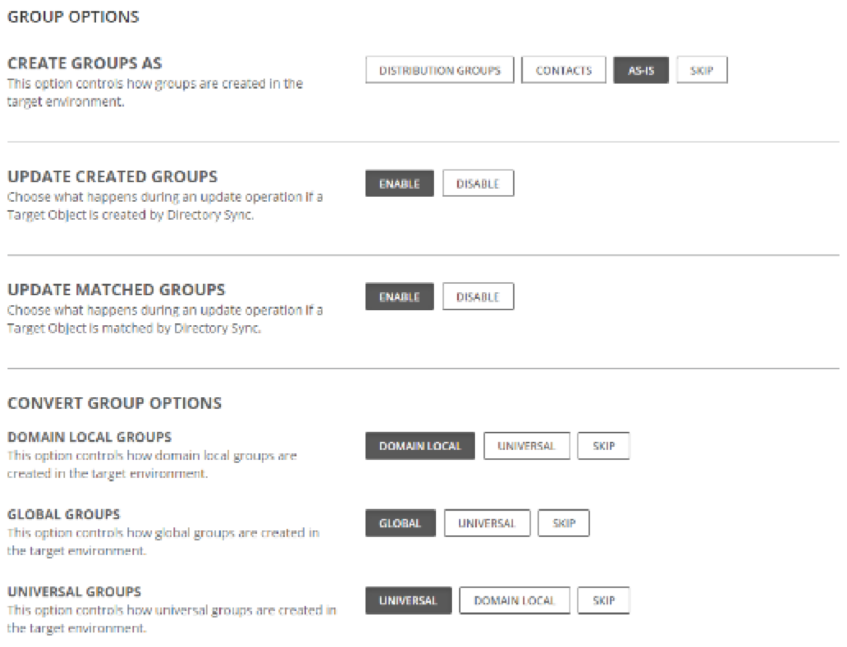
IF TARGET ADDRESS EXISTS
Choose what happens if the *targetAddress* is already set on Target Object

[✓ OVERWRITE ONCE] OVERWRITE ALWAYS DO NOT OVERWRITE

- The allow target address function is used as a condition and should ONLY be used on the targetAddress attribute. The product will determine the correct course of action based on the selection in the settings. This setting can be ignored if you remove AllowTargetAddress() from the condition on the default mapping of the targetAddress.

GetGroupType

- Purpose: Calculate target group type based on profile settings and the source or target object's group type
- Syntax: GetGroupType()
- Location: In the template under Objects – Groups. The possible setting values vary based on the target environment type.



GROUP OPTIONS

CREATE GROUPS AS
This option controls how groups are created in the target environment.

DISTRIBUTION GROUPS CONTACTS [✓ AS-IS] SKIP

UPDATE CREATED GROUPS
Choose what happens during an update operation if a Target Object is created by Directory Sync.

[✓ ENABLE] DISABLE

UPDATE MATCHED GROUPS
Choose what happens during an update operation if a Target Object is matched by Directory Sync.

[✓ ENABLE] DISABLE

CONVERT GROUP OPTIONS

DOMAIN LOCAL GROUPS
This option controls how domain local groups are created in the target environment.

[✓ DOMAIN LOCAL] UNIVERSAL SKIP

GLOBAL GROUPS
This option controls how global groups are created in the target environment.

[✓ GLOBAL] UNIVERSAL SKIP

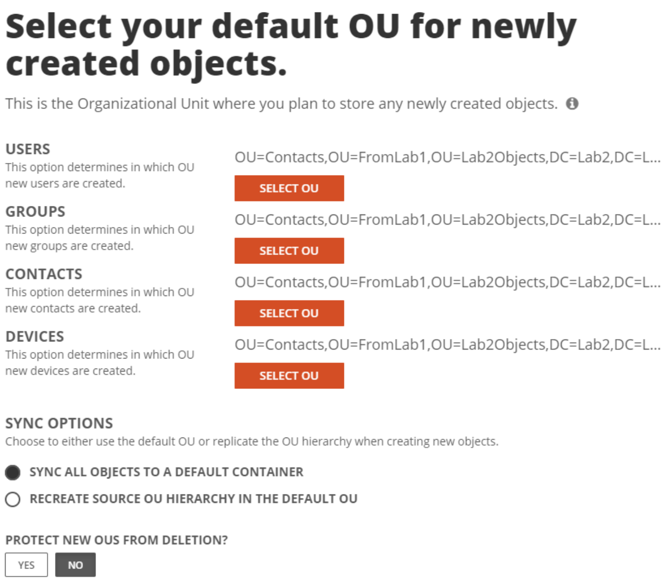
UNIVERSAL GROUPS
This option controls how universal groups are created in the target environment.

[✓ UNIVERSAL] DOMAIN LOCAL SKIP

- GetGroupType() will look at the setting to determine what group type to create in the target. If the target environment is a local Active Directory, it will determine if it should convert the group to another type, skip it, or create it as-is. If the target environment is Azure AD, it will look at the options allowed to determine how you would like the group created.

GetDN

- Purpose: Calculate the target DN value using the given cn value and the value of the target DN
- Syntax: GetDN(cn)
- Location: The default domain


Select your default OU for newly created objects.

This is the Organizational Unit where you plan to store any newly created objects.

USERS
This option determines in which OU new users are created.
	OU=Contacts,OU=FromLab1,OU=Lab2Objects,DC=Lab2,DC=L...
	SELECT OU

GROUPS
This option determines in which OU new groups are created.
	OU=Contacts,OU=FromLab1,OU=Lab2Objects,DC=Lab2,DC=L...
	SELECT OU

CONTACTS
This option determines in which OU new contacts are created.
	OU=Contacts,OU=FromLab1,OU=Lab2Objects,DC=Lab2,DC=L...
	SELECT OU

DEVICES
This option determines in which OU new devices are created.
	OU=Contacts,OU=FromLab1,OU=Lab2Objects,DC=Lab2,DC=L...
	SELECT OU

SYNC OPTIONS

Choose to either use the default OU or replicate the OU hierarchy when creating new objects.

- [✓ SYNC ALL OBJECTS TO A DEFAULT CONTAINER]
- RECREATE SOURCE OU HIERARCHY IN THE DEFAULT OU

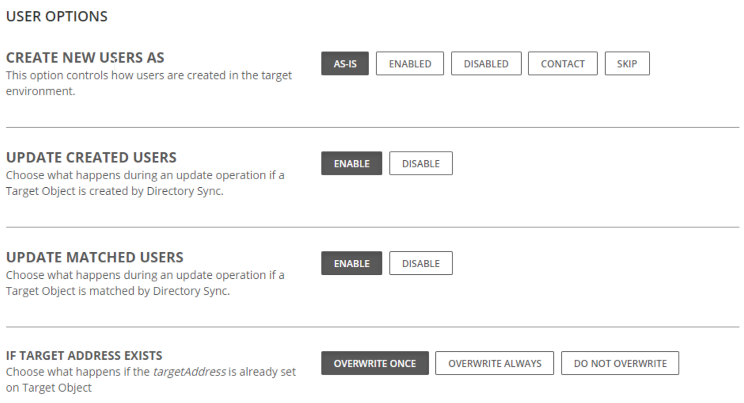
PROTECT NEW OUS FROM DELETION?

YES [✓ NO]

- Based on the above selection and a user with the source CN of john.smith, the DN of the user would be calculated as CN=john.smith,OU=CW-Test,DC=redfishhotels,DC=com

GetUserAccountControl

- Purpose: Calculates UserAccountControl value based on profile settings. Applies to Local AD target only. If the source is a Cloud object, it will read the AccountDisabled Flag and set the object in AD to match the value.
- Syntax: GetUserAccountControl()
- Location: This setting can be in the template under Objects-Users


USER OPTIONS

CREATE NEW USERS AS This option controls how users are created in the target environment.	[✓ AS-IS] ENABLED DISABLED CONTACT SKIP
UPDATE CREATED USERS Choose what happens during an update operation if a Target Object is created by Directory Sync.	[✓ ENABLE] DISABLE
UPDATE MATCHED USERS Choose what happens during an update operation if a Target Object is matched by Directory Sync.	[✓ ENABLE] DISABLE
IF TARGET ADDRESS EXISTS Choose what happens if the *targetAddress* is already set on Target Object	[✓ OVERWRITE ONCE] OVERWRITE ALWAYS DO NOT OVERWRITE

- When using this function only 3 settings from about apply. Enabled, Disabled, and As-Is. The value of UserAccountControl will be set to 512, or 514 based on the values for enabled/disabled, or as it is existing in the source.

General Functions

These functions are used to gather or set values based on other attributes. Some functions will simply return a value and others will return a computed value based on data found within the parameters or complex attribute data.

Lookupvalue

- Purpose: Allow you to lookup a value and replace value with returned value leveraging Data Sets
- Example:
- lookupvalue("TFH-OUS",S.Department,getdn(cn))

encodePWD

- Purpose: Used to encode a string password so that it can be applied to newly created objects in Local AD. This method should not be used to set the password on a cloud object.
- Syntax: EncodePwd(password)
- Example:

Target Attribute	: unicodePwd
Value	: EncodePwd(Profile.DefaultPassword)
Condition	: Action = "create"

GetAccountDisabled

- Purpose: Calculates whether to create a target cloud object as enabled or disabled. This does not apply to local AD environments. If the source is a local AD object, the function will read the UserAccountControl and determine if the object should be enabled or disabled.
- Syntax: GetAccountDisabled()
- Example: This example is also a default mapping for cloud environments. It will set the account as enabled or disabled if we create the object, or update an object that was created by Directory Sync. The limit of only updating objects created by Directory Sync can be removed by removing everything in the condition field.

Target Attribute	: AccountDisabled
Value	: GetAccountDisabled()
Condition	: Action = "create" or (Action = "update" and Target.HasCreateStamp)

ReplaceDomain

- Purpose: Replaces the domain part of the given attribute with the value specified.

- Syntax: ReplaceDomain(attribute, value)
- Attribute can be any attribute that contains an @domain…. Value
- Value specified can be text or anything that represents a domain name.
- Example 1:

  This example replaces the Domain value of the source userPrincipalName with the domain value selected in the Stage Data step of a workflow

  Target Attribute : userPrincipalName

  Value : ReplaceDomain(S.userPrincipalName, Profile.TargetDomain)

  Condition : Null
- Example 2:

  This example replaces the Domain value of the source Target address with the domain value selected in the Stage Data step of a workflow

  Target Attribute : TargetAddress

  Value : ReplaceDomain(S.TargetAddress, Profile.TargetDomain)

  Condition : AllowTargetAddress()

GetProxyAddresses

- Purpose: Create list of Proxy addresses removing any duplicates that are found.
- Syntax: GetProxyAddresses(SourceProxyAddresses,TargetProxyaddresses, address1,Address2,address3)
- Example: Local to Local environment merge Source and Target Proxyaddresses in Target Object

  Target Attribute : Proxyaddresses

  Value : GetProxyAddresses(S.proxyAddresses, T.proxyAddresses, prefix(Result("mail"), "SMTP:"), prefix(legacyExchangeDN, "x500:"), prefix(Result("legacyExchangeDN"), "x500:"))

  Condition :
- Example: Cloud to Cloud environment. SourceProxyAddresses is set to Null by default to avoid errors adding domains which are not accepted by Office 365.

  Target Attribute : Proxyaddresses

  Value : GetProxyAddresses(null, T.EmailAddresses, prefix(Result("WindowsEmailAddress"), "SMTP:"), prefix(LegacyExchangeDN, "x500:"))

  Condition :

GetLegacyExchangeDN

- Purpose: Generate a legacyExchangeDN value using the given exchange org and cn.
- Syntax: GetLegacyExchangeDN(exchangeOrg, cn)

- Example: Set the legacy exchange DN for a user in the target

  Target Attribute : legacyExchangeDN

  Value : GetLegacyExchangeDN(Target.ExchangeOrg, ObjectId)

  Condition : Null

GetObjectClass

- Purpose: Calculates the objectClass from profile settings and the source or target object's class
- Syntax: GetObjectClass()
- Example: Add (contact) to the end of DisplayName if the target object is a contact, if not a contact set same as the source.

  Target Attribute : DisplayName

  Value : if(GetObjectClass()="contact", S.DisplayName+" (Contact)", S.DisplayName)

  Condition :

GetTargetAddress

- Purpose: Calculates targetAddress, by using source primary SMTP, mail, or userPrincipalName
- Syntax: GetTargetAddress()
- Example: Set the target address for a user with the following attributes
- UPN Mail attribute first.m.last@domain.com
- Primary SMTP address first.last@domain.com
- UPN first.last@domain.com

  Target Attribute : targetAddress

  Value : GetTargetAddress()

  Condition :

  This will result in the value of SMTP being used.

GetRecipientTypeDetails

- Purpose: Determines the correct value for RecipientTypeDetails or msExchRecipientTypeDetails
- Syntax: GetRecipientTypeDetails()

Action

- Purpose: Action - returns action currently to be taken for this object ("create", "update", or "delete")
- Syntax: Action = "create"
- Example: Although action could be used within other functions such as case or if, its typical use will be for a condition. In this case we will set the UPN only on create

Target Attribute : userPrincipalName

Value : userPrincipalName

Condition : action = "Create"

result

- Purpose: Returns the calculated mapped value of the given attributeName (must be a string)
- Syntax: result("attributeName")
- Example: Result is used to gather the value of a mapping which contains a function. This example gathers the results of what the function of the "mail" attribute will be the default value/function for the mail attribute is if(GetObjectClass()="group", ReplaceDomain(WindowsEmailAddress, Profile.TargetDomain), WindowsEmailAddress). The alternative for using result("mail") would be to use the entire function which is present in the mail attribute mapping. The result command allows you to link more complex functions.

  Target Attribute : proxyAddresses

  Value :
  GetProxyAddresses(result("mail"), LegacyExchangeDN, result("legacyExchangeDN"))

  Condition : Null

Target.HasCreateStamp

- Purpose : This value is an internal value which determines if Directory Sync created the object being synchronized. This value verifies the "created by Dirsync" text is present on the correct attribute for the target object type.
- Syntax : Target.HasCreateStamp
- Example: You want to set the UserAccountControl value for only objects created by Directory Sync.

  Target Attribute : userAccountControl

  Value : GetUserAccountControl()

  Condition : Action = "create" or (Action = "update" and Target.HasCreateStamp)

ConvertValue("base64STR", objectGUID)

- Purpose: This function is used for syncing ms-DS-ConsistencyGuid, which is an attribute that cannot be directly migrated with a standard mapping. This function can also be used to set the ImmutableID on a cloud object when merging with a local AD object as part of a B2B user conversion process.
- Syntax: ConvertValue("stringtype",objectGUID)
- The only string type currently available is base64STR
- Example: ConvertValue("base64STR", objectGUID)

NewGUID()

- Purpose: This method is used to generate a random GUID
- Syntax: NewGUID()
- Example 1: The following expression will set the target object's CN as a random GUID; this is needed to avoid possible name collision as CN is not a unique attribute in Active Directory.

  Select mapping for ‘DistinguishName and double click, enter the below expression under value field:

  GetDn(NewGuid())
- Example 2: The following expression will set the target object's samAccountName with a random GUID of 20 characters long.

  Select mapping for ‘samAccountName and double click, enter the below expression under value field:

  Left(NewGuid(), 20)

Default Advanced Mappings

The following are the target attributes which contain advanced mapping values by default. The defaults differ based on the source and target environment type.

Note: Changing the default advanced mappings may result in unexpected behavior.

Local to Local

distinguishedName

Target Attribute : distinguishedName

Value : GetDN(cn)

Condition : Null

Definition : This creates the DN for the target user. This is defined by the source CN and by reading the Target OU specified within the Stage Data step of a workflow.groupType

Target Attribute : groupType

Value : GetGroupType()

Condition : Action = "create"

Definition : This will set the group type on creation. The GetGroupType function looks up the setting in the Template selected to determine the target group type.

legacyExchangeDN

Target Attribute : legacyExchangeDN

Value : GetLegacyExchangeDN(Target.ExchangeOrg, objectGuid)

Condition : Null

Definition : This default mapping will read the exchange org info from the target environment and create the LegacyDN using this value.

objectClass

Target Attribute : objectClass

Value : GetObjectClass()

Condition : Action = "create"

Definition : This will read the objectclass of the target user being created based on settings in the template. Create users as, Create Groups as, etc.

proxyAddresses

Target Attribute : proxyAddresses

Value : GetProxyAddresses(result("mail"), legacyExchangeDN, result("legacyExchangeDN"))

Condition : Null

pwdLastSet

Target Attribute : pwdLastSet

Value : 0

Condition : Action = "create"

targetAddress

Target Attribute : targetAddress

Value : GetTargetAddress()

Condition : AllowTargetAddress()

unicodePwd

Target Attribute : unicodePwd

Value : EncodePwd(Profile.DefaultPassword)

Condition : Action = "create"

userAccountControl

Target Attribute : userAccountControl

Value : GetUserAccountControl()

Condition : Action = "create" or (Action = "update" and Target.HasCreateStamp)

userPrincipalName

Target Attribute : userPrincipalName

Value : ReplaceDomain(userPrincipalName, Profile.TargetDomain)

Condition : Null

mail

Target Attribute : mail

Value : if(GetObjectClass()="group", ReplaceDomain(mail, Profile.TargetDomain), mail)

Condition : Null

Cloud to Local

distinguishedName

Target Attribute : distinguishedName

Value : GetDN(cn)

Condition : Null

Definition : This creates the DN for the target user. This is defined by the source CN and by reading the Target OU specified within the Stage Data step of a workflow.

groupType

Target Attribute : groupType

Value : GetGroupType()

Condition : Action = "create"

Definition : This will set the group type on creation. The GetGroupType function looks up the setting in the Template selected to determine the group type.

legacyExchangeDN

Target Attribute : legacyExchangeDN

Value : GetLegacyExchangeDN(Target.ExchangeOrg, objectGuid)

Condition : Null

msExchRecipientTypeDetails

Target Attribute	: RecipientTypeDetails
Value	: GetRecipientTypeDetails()
Condition	: Action = "create"

objectClass

Target Attribute	: objectClass
Value	: GetObjectClass()
Condition	: Action = "create"

proxyAddresses

Target Attribute	: proxyAddresses
Value	: GetProxyAddresses(result("mail"), legacyExchangeDN, result("legacyExchangeDN"))
Condition	: Null

unicodePwd

Target Attribute	: unicodePwd
Value	: EncodePwd(Profile.DefaultPassword)
Condition	: Action = "create"

userAccountControl

Target Attribute	: userAccountControl
Value	: GetUserAccountControl()
Condition	: Action = "create" or (Action = "update" and Target.HasCreateStamp)

userPrincipalName

Target Attribute	: userPrincipalName
Value	: ReplaceDomain(userPrincipalName, Profile.TargetDomain)
Condition	: Null

Cloud to Cloud

objectClass

Target Attribute	: objectClass
Value	: GetObjectClass()
Condition	: Action = "create"

Password

Target Attribute	: Password
Value	: Profile.DefaultPassword
Condition	: Action = "create"

RecipientTypeDetails

Target Attribute	: RecipientTypeDetails
Value	: GetRecipientTypeDetails()
Condition	: Action = "create"

userPrincipalName

Target Attribute	: userPrincipalName
Value	: ReplaceDomain(userPrincipalName, Profile.TargetDomain)

Condition : Null

WindowsEmailAddress

Target Attribute : WindowsEmailAddress

Value : if(GetObjectClass()="group", ReplaceDomain(WindowsEmailAddress, Profile.TargetDomain), WindowsEmailAddress)

Condition : Action = "create"

EmailAddresses

Target Attribute : EmailAddresses

Value : GetProxyAddresses(result("WindowsEmailAddress"), LegacyExchangeDN, null))

Condition : Null

Local to Cloud

AccountDisabled

Target Attribute : AccountDisabled

Value : GetAccountDisabled()

Condition : Action = "create" or (Action = "update" and Target.HasCreateStamp)

groupType

Target Attribute : groupType

Value : GetGroupType()

Condition : Null

Definition : This will set the group type on creation. The GetGroupType function looks up the setting in the Template selected to determine the group type.

objectClass

Target Attribute : objectClass

Value : GetObjectClass()

Condition : Action = "create"

Password

Target Attribute : Password

Value : Profile.DefaultPassword

Condition : Action = "create"

RecipientTypeDetails

Target Attribute : RecipientTypeDetails

Value : GetRecipientTypeDetails()

Condition : Action = "create"

userPrincipalName

Target Attribute	: userPrincipalName
Value	: ReplaceDomain(userPrincipalName, Profile.TargetDomain)
Condition	: Action = "create"

WindowsEmailAddress

Target Attribute	: WindowsEmailAddress
Value	: if(GetObjectClass()="group", ReplaceDomain(mail, Profile.TargetDomain), mail)
Condition	: Action = "create"

EmailAddresses

Target Attribute	: EmailAddresses
Value	: GetProxyAddresses(result("WindowsEmailAddress"), LegacyExchangeDN, null))
Condition	: Null

About us

Quest creates software solutions that make the benefits of new technology real in an increasingly complex IT landscape. From database and systems management, to Active Directory and Office 365 management, and cyber security resilience, Quest helps customers solve their next IT challenge now. Around the globe, more than 130,000 companies and 95% of the Fortune 500 count on Quest to deliver proactive management and monitoring for the next enterprise initiative, find the next solution for complex Microsoft challenges and stay ahead of the next threat. Quest Software. Where next meets now. For more information, visit www.quest.com.

Technical support resources

Technical support is available to Quest customers with a valid maintenance contract and customers who have trial versions. You can access the Quest Support Portal at https://support.quest.com.

The Support Portal provides self-help tools you can use to solve problems quickly and independently, 24 hours a day, 365 days a year. The Support Portal enables you to:

- Submit and manage a Service Request
- View Knowledge Base articles
- Sign up for product notifications
- Download software and technical documentation
- View how-to-videos
- Engage in community discussions
- Chat with support engineers online
- View services to assist you with your product.