

Application Integration (On-Premises) : Integration Server Built-In Services : Utils Folder : pub.utils.messaging:migrateDocTypesTriggersToUM

pub.utils.messaging:migrateDocTypesTriggersToUM

WmPublic. Changes one or more publishable document types to use a Universal Messaging server as the messaging provider.

The `pub.utils.messaging.migrateDocTypesTriggersToUM` service does one or more of the following depending on the provided input:

- Changes the messaging connection alias assigned to the publishable document type to one specified in the service.
- Sets the encoding type of the publishable document type to protocol buffers.
- Synchronizes the updated publishable document types with the messaging provider.
- Converts the filters used by the `webMethods` messaging trigger that subscribe to the publishable document types. Specifically, the service migrates the filter expressions that can be evaluated by the Universal Messaging server from the **Filter** field for a trigger condition to the **Provider Filter (UM)** field. This may include some syntax changes.

Input Parameters

reportOnly

java.lang.Boolean Optional. Indicates whether the service changes the document types and `webMethods` messaging triggers and returns a summary of the changes or if the service returns only a summary without making any changes. Set to:

- `true` to return a summary of the document types and `webMethods` messaging trigger that the service would successfully and unsuccessfully update, including the errors and warnings that would be encountered. The service does not modify any document types and triggers. This is the default.
- `false` to update the publishable document types and `webMethods` messaging triggers, synchronize the publishable document types with the messaging provider, and return a summary of the changes.

backupPackages

java.lang.Boolean Optional. Indicates whether the service creates a package archive for each package affected by this service prior to making any updates to the publishable document types and `webMethods` messaging trigger. Set to:

- `true` to create an archive for each package that contains a publishable document type or `webMethods` messaging trigger that will be updated by this service. The archive will be named *packageName.zip_timeStamp* and placed in the following directory: and will be located in the following directory: *Integration Server_directory\instances\instance_name\replicate\archive*. This is the default.
- `false` to skip creating an archive of each affected package before making updates.

Important: Software AG recommends that you set *backupPackages* to true when *reportOnly* is set to false if you have not taken other measures to back up the packages containing the publishable document types and triggers to be migrated prior to executing the `pub.utils.messaging:migrateDocTypesTriggersToUM` service. After the service executes successfully, restoring from a backup file is the only way to revert a change made by the service.

packageNames

String List Optional. Name of the packages containing the publishable document types that you want to migrate. Integration Server also migrates any `webMethods` messaging triggers in packages that reference publishable document types contained in the specified packages.

The *packageNames* and *documentTypesNames* input parameters are mutually exclusive. If you specify a value for both parameters, the service fails with an exception. If you do not specify a value for either, the service updates publishable document types and `webMethods` messaging triggers in all packages that do not start

with the letters “wm” in any case combination.

documentTypeNames

String List Optional. Fully qualified names of the publishable document types that you want to update.

The *packageNames* and *documentTypeNames* input parameters are mutually exclusive. If you specify a value for both parameters, the service fails with an exception. If you do not specify a value for either, the service updates publishable document types and webMethods messaging triggers in all packages that do not start with the letters “wm” in any case combination.

newConnectionAlias

String Optional. Name of the Universal Messaging (UM) connection alias to assign to the publishable document types. Any webMethods messaging trigger that subscribe to updated publishable document types will use this alias to receive published documents.

You must specify a *newConnectionAlias* to update any publishable document types that currently specify the Broker connection alias or are publishable locally only. If you do not specify a *newConnectionAlias*, the service skips these publishable document types.

If the default connection alias is the Broker connection alias and you do not specify a *newConnectionAlias*, the service skips any publishable document types that use the default connection alias. However, if you change the default connection alias to be Universal Messaging connection alias and then execute the `pub.utils.messaging:migrateDocTypesTriggersToUM` service, the service migrates any publishable document types that use the default messaging connection alias as well as any triggers that subscribe to those publishable document types.

If a publishable document types already specifies a Universal Messaging connection alias, you do not need to specify a *newConnectionAlias*. However, you can use the service to change the Universal Messaging connection alias used by one or more publishable document types.

Output Parameters

affectedPackages

Document Conditional. An IData containing key=value pairs that identify the packages impacted by the migration and the name of the archive created for the package. The service returns an archive name only if *reportOnly* is set to false and *backupPackages* is set to true. The archive will be named *packageName.zip_timeStamp* and will be located in the following directory: *Integration Server_directory\instances\instance_name\replicate\archive*.

The service returns *affectedPackages* only if the service finds at least one publishable document type or webMethods messaging trigger to update.

failedPackages

Document List Conditional. List of the packages that could not be archived. If Integration Server cannot create a package archive for any of the affected packages, the service ends with an error and nothing is migrated.

The service attempts to archive a package only when *backupPackages* is set to true.

The service returns the *failedPackages* parameter only if one of the following occurs:

- The *backupPackages* input parameter is set to true and an error prevented the service from archiving at least one package.
- A package listed in the *packageNames* input parameter did not contain any publishable document types or webMethods messaging triggers to update.

Key

Description

package

String Name of the package that the service could not archive because of an error.

errors **String** Text of the error message that caused package archiving to fail.

updatedDocumentTypes

Document List Conditional. List of the publishable document types updated by the service.

The service returns the *updatedDocumentTypes* parameter only if the service updated one or more publishable document types.

Key	Description
<i>package</i>	String Name of the package containing the updated publishable document type.
<i>name</i>	String Fully qualified name of the updated publishable document type.

skippedDocumentTypes

Document List Conditional. List of the publishable document types that were not updated because there were no changes to make. Some reasons the service might skip a publishable document type include, but are not limited to:

- The publishable document type already uses the messaging connection alias specified in *newConnectionAlias* and the encoding type for the publishable document type is already set to protocol buffers.
- The publishable document type uses a Broker connection alias and a Universal Messaging connection alias was not specified in *newConnectionAlias*.
- Values were not specified for either the *packageNames* or the *documentTypeNames* input parameters and Integration Server found publishable document types in packages whose names started with the letters "wm" in any case combination

The service returns the *skippedDocumentTypes* parameter only if the service skipped updating one or more publishable document types.

Key	Description
<i>package</i>	String Name of the package containing the skipped publishable document type.
<i>name</i>	String Fully qualified name of the skipped publishable document type.
<i>reason</i>	String Message indicating why the service did not update the publishable document type.

failedDocumentTypes

Document List Conditional. List of the publishable document types that the service could not update because of an error.

The service returns the *failedDocumentTypes* parameter only if there were one or more publishable document types that the service could not update because of an error.

Key	Description
<i>package</i>	String Name of the package containing the publishable document type that the service could not update because of an error.
<i>name</i>	String Name of the publishable document type that the service could not update because of an error.
<i>errors</i>	String Text of the error message that prevented the service from updating the publishable document type.

*providerSync
Results*

Document List Conditional. Results of synchronizing the updated publishable document types with the messaging provider.

The service only produces this output parameter when *reportOnly* is set to false and the service updated at least one publishable document type.

Key	Description
<i>successfulPDTs</i>	String List Fully qualified names of the publishable document types that the service synchronized successfully with the messaging provider.
<i>unsuccessfulPDTs</i>	String List Fully qualified names of the publishable document types that the service did not synchronize successfully with the messaging provider.
<i>errors</i>	Document List List of any errors that occurred during document type synchronization.
<i>warnings</i>	Document List List of any warnings that occurred during document type synchronization

updatedTriggers

Document List Conditional. List of the webMethods messaging triggers updated without warnings by the service. When warnings do not occur when updating the trigger, it indicates that Integration Server migrated the entire contents of the **Filter** field to the **Provider Filter (UM)** field for all the conditions in the trigger.

The service returns the *updatedTriggers* parameter only if the service updated at least one trigger without warnings.

Key	Description
<i>package</i>	String Name of the package containing the updated webMethods messaging trigger.
<i>name</i>	String Fully qualified name of the updated webMethods messaging trigger.
<i>reason</i>	String Identifies any filters that were converted. For a trigger that was updated without warnings, <i>reason</i> displays the contents of the Filter field prior to migration of the trigger filter, the contents of the Provider Filter (UM) field after migration of the trigger filter, and an indication that the filter was converted successfully.

*updatedTriggers
WithWarning*

Document List Conditional. List of the webMethods messaging triggers updated but with warnings. Warnings indicate that the service could not migrate the entire contents of the **Filter** field to the **Provider Filter (UM)** field for all the conditions in the trigger. Integration Server might have migrated some expressions in the trigger conditions, but not the entire filter.

When a trigger is updated with warnings, the service keeps the original filter in the **Filter** field and places the filter expressions that were migrated in the **Provider Filter (UM)** field

The service returns the *updatedTriggersWithWarning* parameter only if the service updated at least one trigger for which a warning occurred.

Key	Description
<i>package</i>	String Name of the package containing the updated webMethods messaging trigger.
<i>name</i>	String Fully qualified name of the updated webMethods messaging trigger.

<i>reason</i>	String Identifies any filters that were converted. For a trigger that was updated with warnings, <i>reason</i> displays the expressions in the Filter field that were migrated, the converted expression in the Provider Filter (UM) , and an indication that the filter was converted successfully.
<i>warning</i>	String Warning message indicating why the complete original filter in the Filter field could not be migrated to the Provider Filter (UM) field.

skippedTriggers

Document List Conditional. List of the webMethods messaging triggers that the service did not update because no changes were needed.

The service returns the *skippedTriggers* parameter only if the service skipped updating at least one trigger.

Key	Description
<i>package</i>	String Name of the package containing the skipped webMethods messaging trigger.
<i>name</i>	String Fully qualified name of the skipped webMethods messaging trigger.
<i>reason</i>	String Message indicating why the service did not make any updates for the webMethods messaging trigger. For example, if the trigger conditions did not contain any filters, the reason will be "Trigger did not contain any filters."

failedTriggers

Document List Conditional. List of the webMethods messaging triggers that the service could not update because of an error.

The service returns the *failedTriggers* parameter only if at least one trigger failed migration.

Key	Description
<i>package</i>	String Name of the package containing the webMethods messaging trigger that the service could not update because of an error.
<i>name</i>	String Name of the webMethods messaging trigger that the service could not update because of an error.
<i>errors</i>	String Text of the error message that prevented the service from updating the webMethods messaging trigger

Usage Notes

To execute the `pub.utils.messaging.migrateDocTypesTriggersToUM`, Integration Server must be connected to a Broker. Integration Server must have a Broker connection alias that configures a connection to the Broker that contains the provider definitions for the publishable document types and the trigger subscriptions that you want to updated. The Broker connection alias must be enabled at the time the `pub.utils.messaging.migrateDocTypesTriggersToUM` executes.

Make sure the publishable document types that use a Broker connection alias are synchronized with the associated Broker document types prior to executing the `pub.utils.messaging.migrateDocTypesTriggersToUM` service. You can synchronize publishable document types using Designer or by executing the `pub.publish:syncToProvider` service.

When using Universal Messaging as the messaging provider, webMethods messaging trigger require more server threads and system threads than needed when using Broker as the messaging provider. Before migrating publishable document types and webMethods messaging triggers to work with Universal Messaging, Software AG recommends increasing the size of your server thread pool by at least two additional threads per trigger that will be migrated. For example, if you intend to migrate 100 triggers, increase the server thread pool by 200 threads.

As part of updating publishable document types and webMethods messaging triggers to work with Universal Messaging, first run

the `pub.utils.messaging.migrateDocTypesTriggersToUM` service in report only mode, that is with `reportOnly` set to true. Use the service output to find potential problems in the migration, such as identifying publishable document types and triggers that will not be migrated successfully. When you are confident that all the publishable document types and webMethods messaging triggers that you want to migrate will do so successfully, run the `pub.utils.messaging.migrateDocTypesTriggersToUM` service is with `reportOnly` set to false.

Prior to executing the `pub.utils.messaging:migrateDocTypesTriggersToUM` service, if you have not taken other measures to back up the packages containing the publishable document types and triggers to be migrated, Software AG recommends that you set `backupPackages` to true when `reportOnly` is set to false. After the service executes successfully, restoring from a backup file is the only way to revert a change made by the service.

Publishable document types that use a Universal Messaging connection alias cannot be published locally. Do not migrate a publishable document type that is used in local publishing.

When determining which publishable document types to migrate, make sure that you migrate all of the document types subscribed to by a group of webMethods messaging triggers at one time. This ensures that the webMethods messaging triggers will subscribe to document types that use the same messaging connection alias. Integration Server considers a trigger to be invalid if it subscribes to publishable document types that use different messaging connection aliases. The `pub.utils.messaging.migrateDocTypesTriggersToUM` service will save a webMethods messaging trigger that uses mixed connection aliases but will disable the trigger. Additionally, the service will not attempt to convert any of the filters in conditions of the disabled trigger. To resolve this issue, you can modify the publishable document types used by the trigger so that the publishable document types all use the same messaging connection alias. You can then re-execute the `pub.utils.messaging.migrateDocTypesTriggersToUM` service to migrate the encoding type of the publishable document types as well as the trigger filters. Alternatively, you can restore the publishable document types that the trigger subscribes to by replacing the migrated packages with the archive packages created by the service when `backupPackages` is set to true. For example, suppose that trigger `triggerAB` subscribes to `documentA` and `documentB` both of which use a Broker connection alias. In `documentTypeNames`, you specify only `documentA`. When the service executes it migrates `documentA`. However, the service does not migrate `triggerAB` because the trigger subscribes to publishable document types that use different messaging connection aliases. This configuration is invalid. The service saves `triggerAB` in a disabled state.

If you do not specify a `newConnectionAlias`, the publishable document type already uses a Universal Messaging connection alias, and the publishable document type has an encoding type of `IData`, the `pub.utils.messaging.migrateDocTypesTriggersToUM` service changes the encoding type of the publishable document type to protocol buffers. The service then updates the filters for any webMethods messaging triggers that subscribe to the document types so that the filters are compatible with protocol buffers.

If publishable document type use the default messaging connection alias and the default messaging connection alias is the Broker connection alias, you can migrate your document types and triggers to work with Universal Messaging and use the protocol buffer encoding type by first switching the default connection alias to a Universal Messaging connection alias and then running the `pub.utils.messaging:migrateDocTypesTriggersToUM` service with `reportOnly` set to false. Use Integration Server Administrator to change the default messaging connection alias. Note that a publishable document type uses the default messaging connection alias if the **Connection alias name** property for the document type is set to `DEFAULT`.

The `pub.utils.messaging:migrateDocTypesTriggersToUM` service automatically attempts to migrate the webMethods messaging triggers that subscribe to the publishable document types updated by the service. Migrating a trigger consists of migrating a filter. To migrate a filter, the service does the following:

1. The `pub.utils.messaging:migrateDocTypesTriggersToUM` service examines the expressions in the **Filter** field in a trigger condition and determines which expressions can become a provider filter on Universal Messaging. Some filter expressions that are valid as local filters on Integration Server or valid as filters on Broker contain syntax that is not supported by protocol buffers or Universal Messaging
2. The `pub.utils.messaging:migrateDocTypesTriggersToUM` service migrates expressions that are valid filters for protocol buffers to the **Provider Filter (UM)** field.
3. The service then does one of the following:
 - If the `pub.utils.messaging:migrateDocTypesTriggersToUM` service can migrate all the expressions in the **Filter** field to the **Provider Filter (UM)** field, the service deletes the contents of the **Filter** field. The service lists the trigger in the `updatedTriggers/name` output parameter. The service lists the successfully migrated filter expressions in `updatedTriggers/reason` output parameter.
 - If the `pub.utils.messaging:migrateDocTypesTriggersToUM` service can migrate only some of the expressions in the **Filter** field, the service migrates the filter expressions that it can in the **Provider Filter (UM)** field. The service leaves the original filter in the **Filter** field. The service lists the trigger in the `updatedTriggersWithWarning/name` output parameter. The service lists the successfully migrated filter expressions in `updatedTriggersWithWarning/reason` output parameter and lists the filter expressions that it could not migrate in the `updatedTriggersWithWarning/warning` output parameter.

For a webMethods messaging trigger that receives messages from Universal Messaging, the contents of the **Filter** field are used as a local filter. Integration Server applies the local filter after it receives the message from Universal Messaging

- If the `pub.utils.messaging:migrateDocTypesTriggersToUM` service cannot migrate any of the expressions in the **Filter** field, the service leaves the original filter in the **Filter** field. The service lists the trigger in the *updatedTriggersWithWarning/name* output parameter. The service lists the filter expressions that it could not migrate in the *updatedTriggersWithWarning/warning* output parameter.

The `pub.utils.messaging:migrateDocTypesTriggersToUM` updates publishable document types contained in packages that begin with “Wm” when the document type is specified in *documentTypeNames* or the package is specified in *packageNameNames* only.

The `pub.utils.messaging:migrateDocTypesTriggersToUM` service synchronizes updated publishable document types with the provider definition on the messaging provider. Be sure to review the contents of *providerSyncResults/errors* and *providerSyncResults/warnings* for errors and warnings that Integration Server generated during synchronization

If Integration Server did not successfully synchronize an updated publishable document type use the returned errors in *providerSyncResults/errors* to determine why Integration Server could not synchronize the publishable document type. Then, fix the cause of the error and synchronize the publishable document type with the messaging provider. You can synchronize documents using Designer or by executing the `pub.publish:syncToProvider` service.

The `pub.utils.messaging:migrateDocTypesTriggersToUM` service will not modify any publishable document types or webMethods messaging triggers that are locked for edit by another user or checked out at the time the service executes.

To use the `pub.utils.messaging:migrateDocTypesTriggersToUM` service to update a publishable document type and webMethods messaging trigger, the user or client invoking the service must have Write access to the publishable document type and webMethods messaging triggers.

Use the *webMethods Error Message Reference* to find out more information about the error and warning messages returned by `pub.utils.messaging:migrateDocTypesTriggersToUM` service.

To generate additional logging information in the Sever log when the `pub.utils.messaging:migrateDocTypesTriggersToUM` service executes, set the logging level for the server log facilities **0153 Dispatcher (Universal Messaging)** and **0154 Protocol Buffer Encoding (Universal Messaging)** to Debug or Trace. Increased logging can help you to locate problems that occur during service execution.

Copyright © 2015- 2016 Software AG, Darmstadt, Germany.

[Contact Support](#) | [Community](#) | [Feedback](#)