

webMethods EntireX

IDL Extractor for Integration Server

Innovation Release

Version 9.9

October 2015

This document applies to webMethods EntireX Version 9.9 and all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 1997-2015 Software AG, Darmstadt, Germany and/or Software AG USA, Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA, Inc. and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at <http://softwareag.com/licenses>.

Use of this software is subject to adherence to Software AG's licensing conditions and terms. These terms are part of the product documentation, located at <http://softwareag.com/licenses/> and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at <http://softwareag.com/licenses> and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software AG.

Document ID: EXX-EEXXAIETRACTOR-99-20171128

Table of Contents

Software AG IDL Extractor for webMethods Integration Server	v
1 Introduction to the IDL Extractor for Integration Server	1
Scope	2
Calling an Integration Server Flow from a Mainframe Program	3
2 Using the IDL Extractor for Integration Server	5
Step 1: Start the IDL Extractor for Integration Server	6
Step 2a: Connect EntireX Workbench with Integration Server	7
Step 2b: Connect EntireX Workbench with Integration Server using an Existing Connection	8
Step 3: Select the Integration Server Package to Extract	9
Step 4: Select the Connection Type	12
Step 5: Define a Listener	13
3 Mapping Integration Server Data Types to IDL	15
4 Using the Service <code>pub.wmentirex.listener:generateIDLfromService</code>	19
5 Integration Server Preferences	21
Integration Server Connections	22
Setting Integration Server Preferences	24

Software AG IDL Extractor for webMethods Integration Server

The Software AG IDL Extractor for webMethods Integration Server is a wizard that reads a package from the Integration Server and generates a Software AG IDL file from all the existing services and nodes. Each service results in a program in the IDL file. All parameters of the services are mapped to an IDL alphanumeric data type, available as variable (AV) or fixed (An) length.

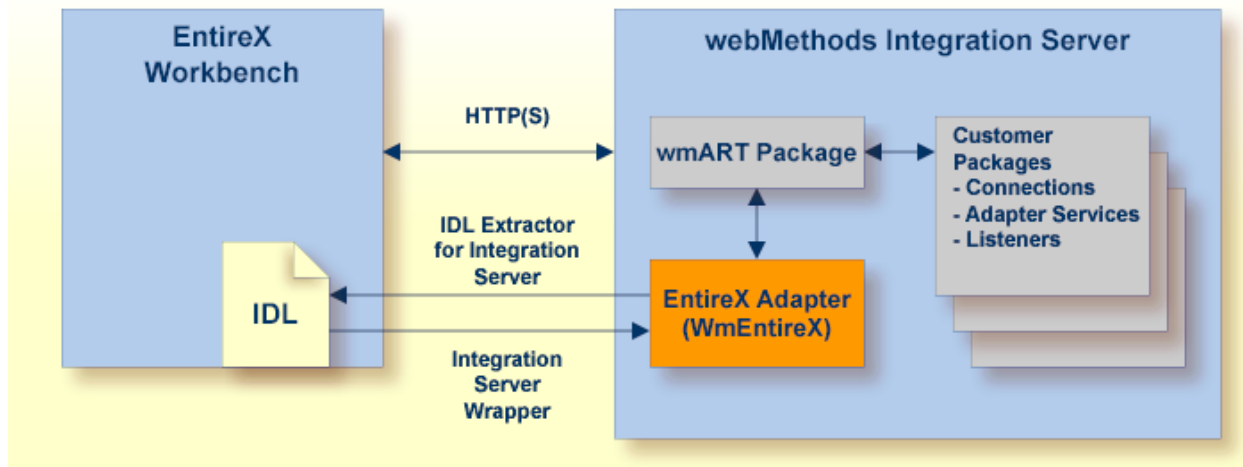
Introduction	Introduction to the IDL Extractor for Integration Server.
Using	Using the IDL Extractor for Integration Server.
<i>IS to IDL Mapping</i>	Mapping Integration Server data types and syntax to Software AG IDL.
<i>Automating IDL Generation</i>	Creating an IDL file can be automated using service <code>pub.wmentirex.listener:generateIDLfromService</code> .
Preferences	Describes the Integration Server preferences.

1 Introduction to the IDL Extractor for Integration Server

- Scope 2
- Calling an Integration Server Flow from a Mainframe Program 3

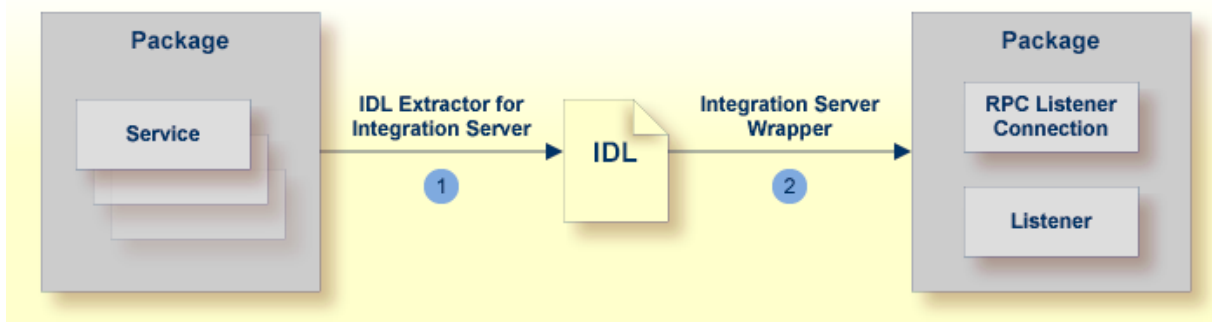
Scope

The Software AG IDL Extractor for webMethods Integration Server is a wizard that reads a package from the Integration Server and generates a Software AG IDL file from all the existing services and nodes. Each service results in a program in the IDL file. All parameters of the services are mapped to an IDL alphanumeric data type, available as variable (AV) or fixed (An) length.



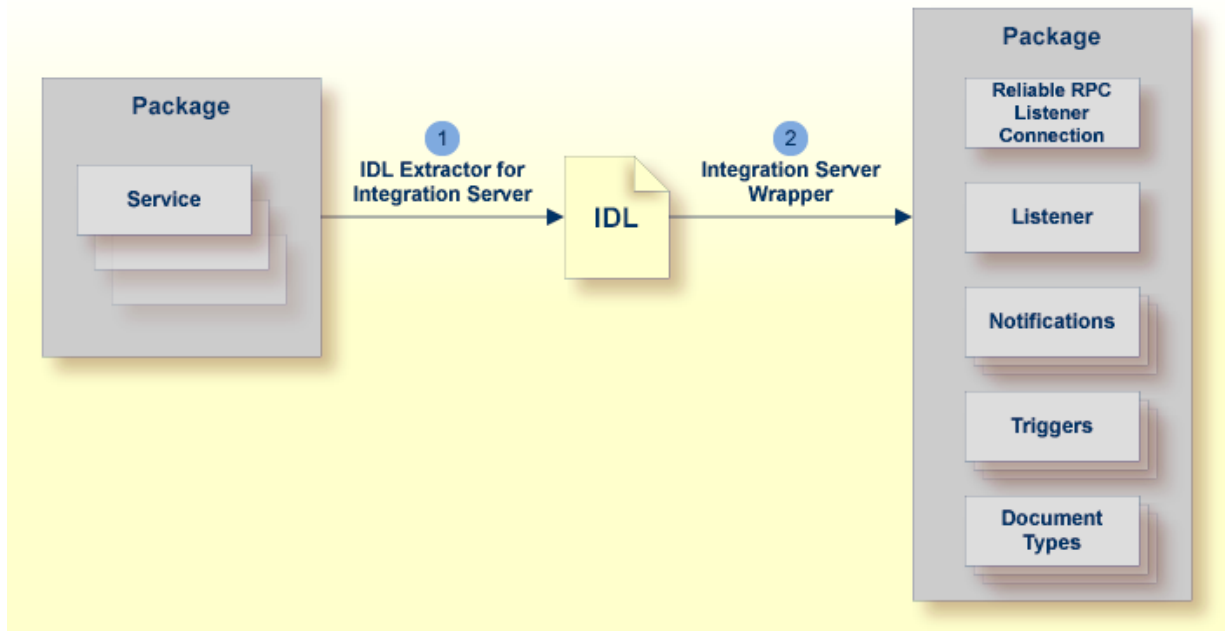
With the Integration Server Wrapper you can generate a server of type "RPC Listener" or "Reliable RPC Listener". See *Using the Integration Server Wrapper*.

■ RPC Listener



1. See [Using the IDL Extractor for Integration Server](#).
2. See [Step 4b: Define Adapter Services for an RPC Listener or a Reliable RPC Listener](#).

■ Reliable RPC Listener



1. See [Using the IDL Extractor for Integration Server](#).
2. See [Step 4b: Define Adapter Services for an RPC Listener or a Reliable RPC Listener](#).

Calling an Integration Server Flow from a Mainframe Program

➤ To call an Integration Server flow from a mainframe program

- Use the plug-in to generate the connection and see [Using the Integration Server Wrapper](#) with the steps 1 to 3, 4b and 5.


2 Using the IDL Extractor for Integration Server

▪ Step 1: Start the IDL Extractor for Integration Server	6
▪ Step 2a: Connect EntireX Workbench with Integration Server	7
▪ Step 2b: Connect EntireX Workbench with Integration Server using an Existing Connection	8
▪ Step 3: Select the Integration Server Package to Extract	9
▪ Step 4: Select the Connection Type	12
▪ Step 5: Define a Listener	13

Step 1: Start the IDL Extractor for Integration Server

➤ To start the IDL Extractor for Integration Server

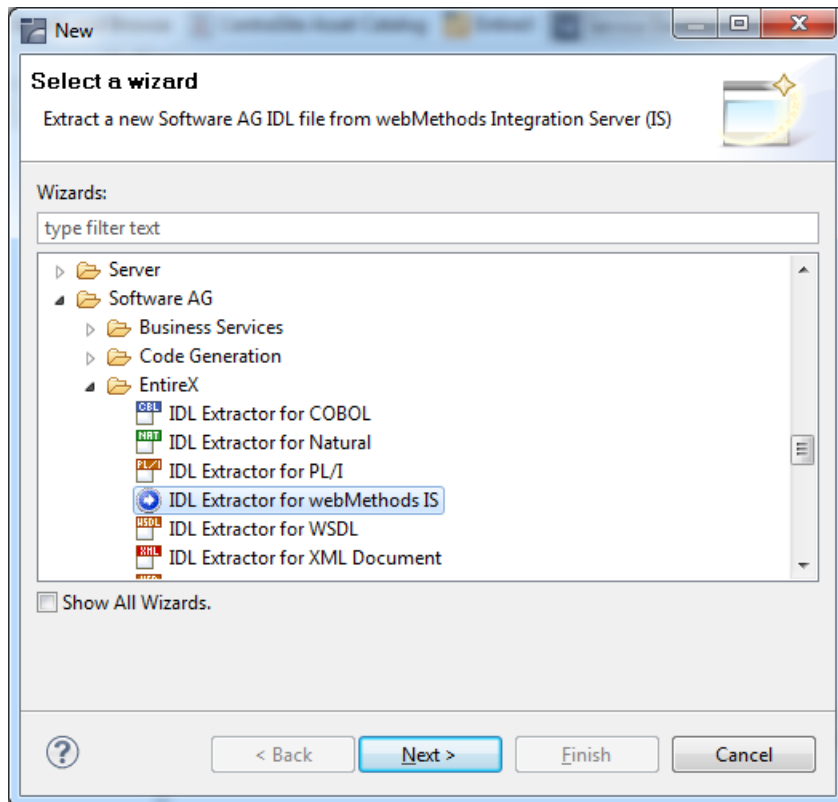
- 1 The IDL Extractor is a *New Wizard* in Eclipse. Choose **New** from the file menu, select **IDL Extractor for webMethods IS** in the following page and choose **Next**.

 **Note:** You can also choose **New** from the toolbar or context menu of a view showing resources. Also **Ctrl-N** starts the selection of the *New Wizards*.

- 2 If you are using the wizard for the first time without any predefined Integration Server connections, continue with [Step 2a: Connect EntireX Workbench with Integration Server](#).

Or:

If Integration Server connections are already defined (see *Integration Server Connections*), or you want to communicate with an additional Integration Server, continue with [Step 2b: Connect EntireX Workbench with Integration Server using an Existing Connection](#).



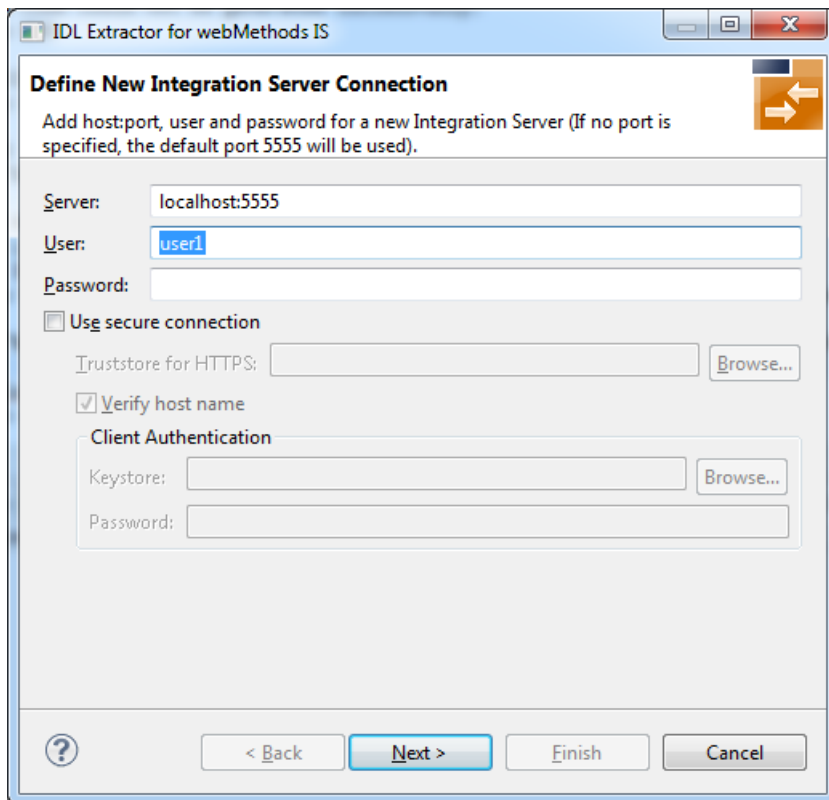
Step 2a: Connect EntireX Workbench with Integration Server

➤ To create a new Integration Server connection

- 1 Define the new Integration Server connection on the following wizard page.

 **Notes:**

1. The only required field is **Server**. Enter the hostname of the Integration Server including an optional port number. If no port number is specified, port number defaults to "5555". The **Integration Server Authentication** can be passed with the **User** and **Password** fields.
2. Optional settings are for secure connections. The **Truststore for HTTPS** contains all signed certificates and must be a valid truststore.
3. The check box **Verify host name** checks that the hostname is entered in the stored certificate.
4. When the Integration Server has **Client Authentication** enabled, you can specify your **Keystore** file and keystore **Password**.
5. For managing Integration Server connections, see *Integration Server Preferences*.



Define New Integration Server Connection

Add host:port, user and password for a new Integration Server (if no port is specified, the default port 5555 will be used).

Server: localhost:5555

User: user1

Password:

Use secure connection

Truststore for HTTPS: Browse...

Verify host name

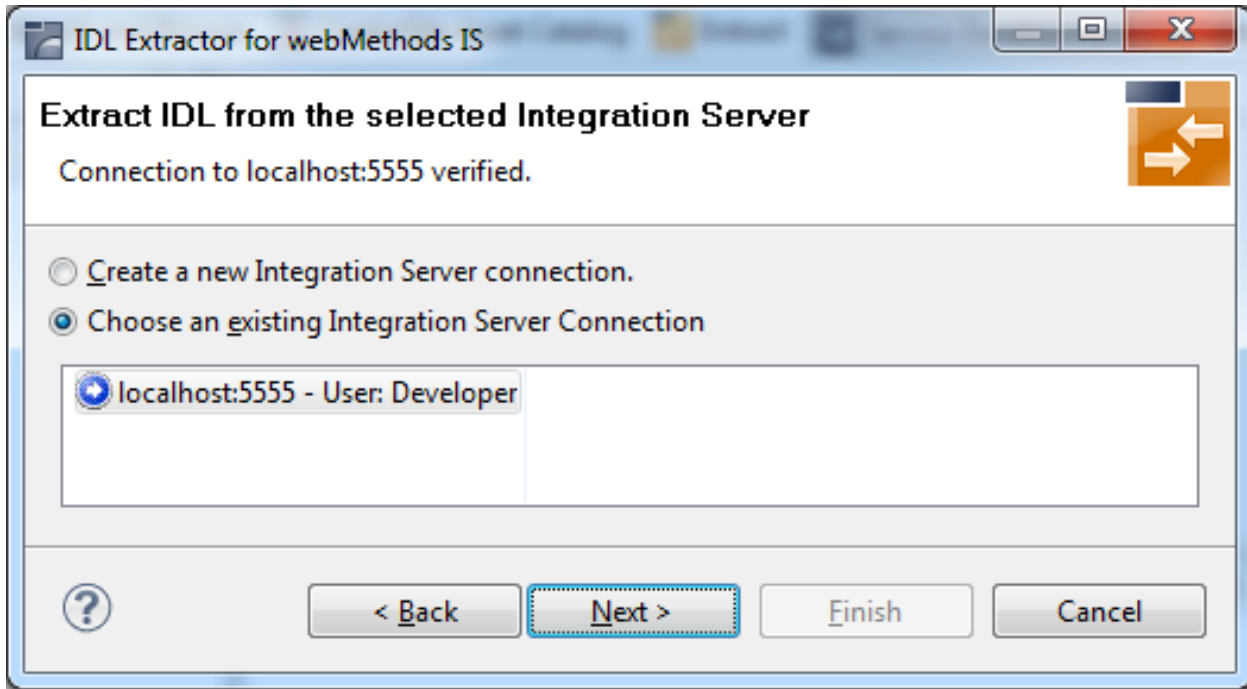
Client Authentication

Keystore: Browse...

Password:

- 2 Choose **Next** and continue with [Step 3: Select the Integration Server Package to Extract](#).

Step 2b: Connect EntireX Workbench with Integration Server using an Existing Connection



➤ To use an existing Integration Server connection

- 1 Select **Choose an existing Integration Server Connection** and an Integration Server connection from the list.

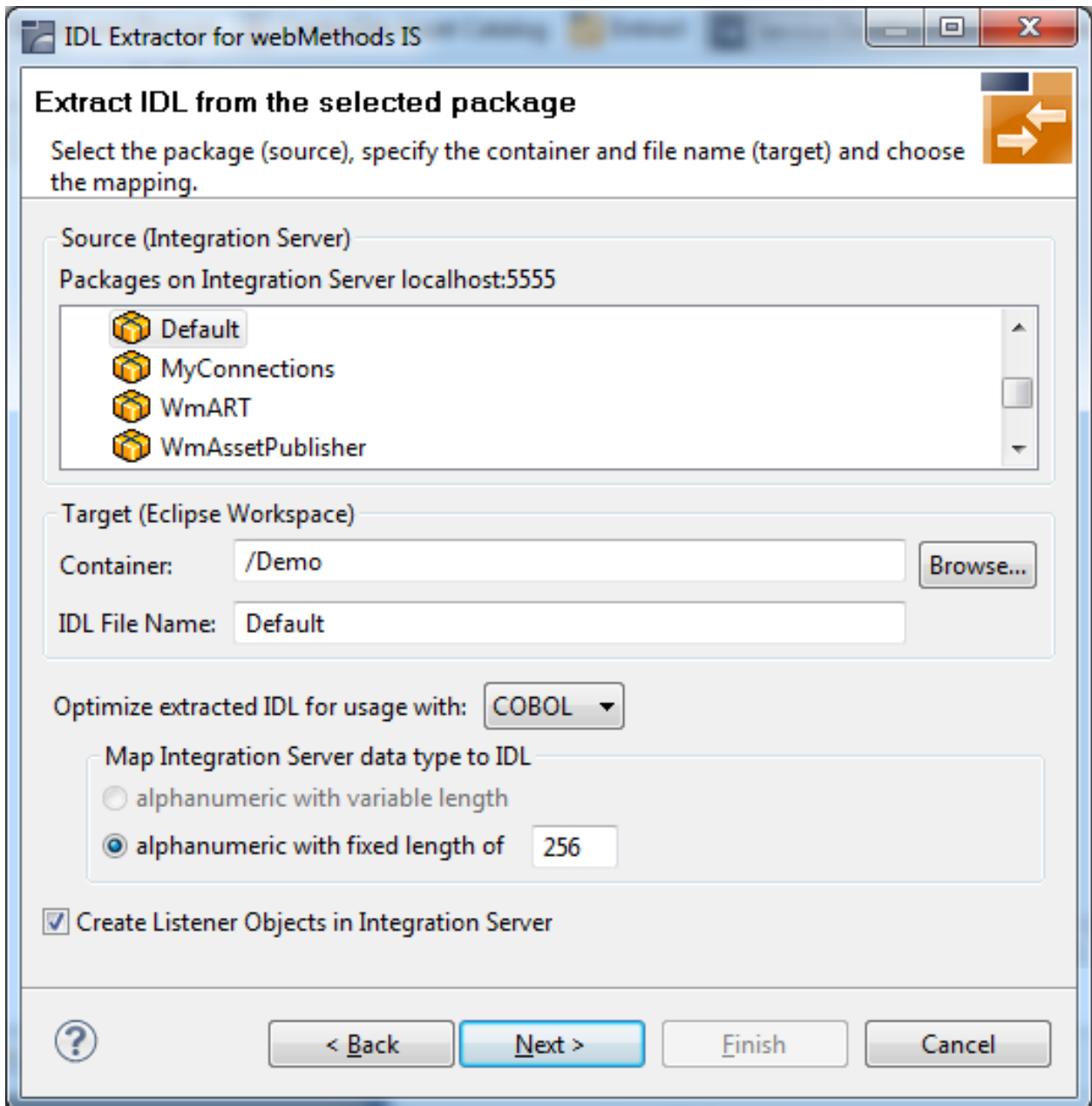
The selected connection is verified by a ping command. If the response is valid, the **Next** button is enabled. If invalid, an error message is displayed.

- 2 Continue with *Step 3: Select the Integration Server Package to Extract.*

➤ To create an additional Integration Server connection

- Select **Create a new Integration Server Connection** and continue with *Step 2a: Connect EntireX Workbench with Integration Server.*

Step 3: Select the Integration Server Package to Extract



➤ To extract the IDL from the selected package

- 1 Select the package to extract (from the list indicated by **Source**), for example the package "Default" in the screen above.
- 2 Specify the target. By default, the wizard tries to find a valid container based on your position in the Navigator or Package Explorer View.

Or:

Choose **Browse...** to select a container from your workspace.



Notes:

1. The IDL file name is based on the selected Integration Server package from the list below, but this is only a proposal and can be changed.
 2. If the file name already exists in your container, a warning will be displayed in the title area of the wizard page.
 3. If the extension "idl" is not specified, it will be added automatically.
- 3 Optimize the extracted IDL. Use the combo-box and select the target programming language:

■ **COBOL**

For usage with the *COBOL Wrapper*. Enter the *default length* for IDL type `A(default-length)` fields which map then to COBOL alphanumeric data items `PIC X(default-length)`.

■ **Natural**

For usage with the *Natural Wrapper*. Choose one option indicated by **Map Integration Server data type to IDL:**

- alphanumeric with variable length: Extract to IDL Type `AV` fields which map then to Natural `DYNAMIC` data types. The interface does not have any length restrictions.
- alphanumeric with fixed length: Extract to IDL Type `A(default-length)` fields which map then to Natural fixed-length alphanumeric data types. The interface from a Natural point of view is more legacy-like and easier to program, but there are length restrictions.

■ **PL/I**

For usage with the *PL/I Wrapper*. Enter the *default length* for IDL Type `A(default-length)` fields which map then to PL/I fixed-length alphanumeric fields `CHAR (default-length)`.

■ **Other**

For usage with other *EntireX Wrappers*. If you later use wrappers other than the COBOL Wrapper, Natural Wrapper or PL/I Wrapper, no options need to be specified.

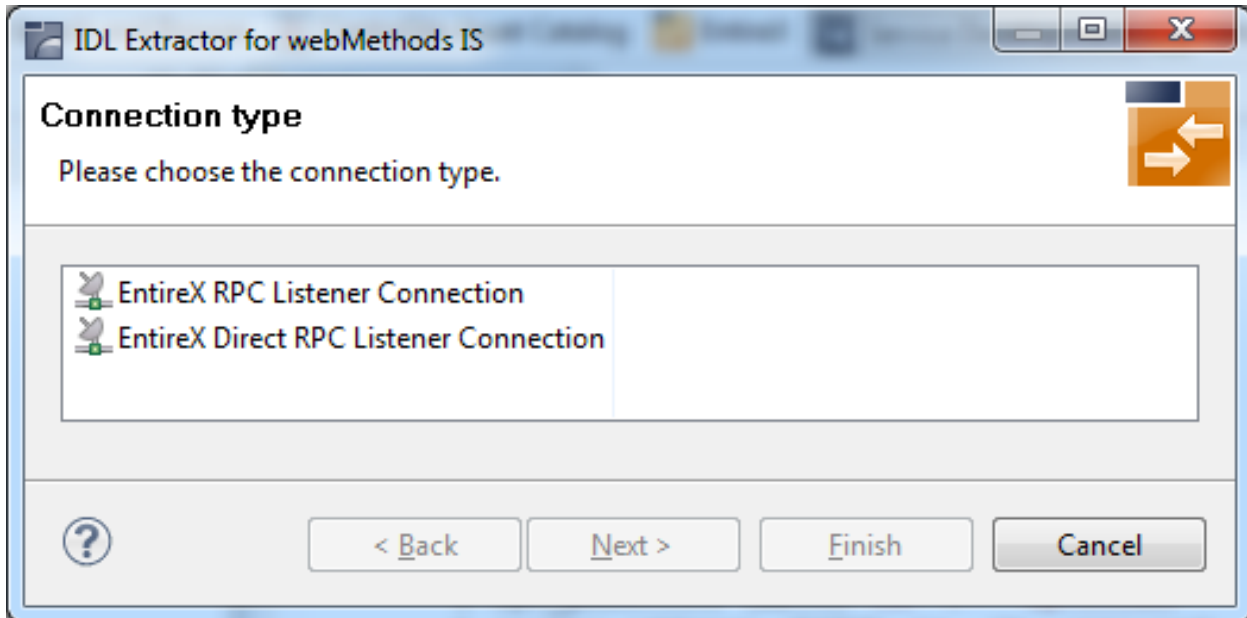
- 4 Clear **Create Listener Objects in Integration Server** to only extract the IDL file from the package. The wizard finishes after extracting the IDL.



Tip: This is useful if you want to modify the IDL (manually) before generating clients. In this case, the connections and listeners have to be created in a separate step. See *Step 4b: Define Adapter Services for an RPC Listener or a Reliable RPC Listener* in the Integration Server Wrapper documentation.

- 5 Choose **Next** to store the IDL file in the selected container in the Eclipse workspace. When you go to the next page, the **Back** button is disabled, because the IDL file has already been created and this step cannot be reverted.

Step 4: Select the Connection Type



> To select the connection type

As a prerequisite, the IDL file has been stored in the workspace and the **Back** button is now disabled.

- Select a **Connection type** and click **Next**.

 **Notes:**

1. An EntireX RPC Listener connection is always available.
2. An EntireX Reliable RPC Listener connection is available if all IDL programs contain only `IN` parameters.
3. An EntireX Direct RPC Listener connection is available if it is enabled by the license for the webMethods EntireX Adapter.

Step 5: Define a Listener

Define Adapter Services for EntireX RPC Listener Connection
Select a package, name a folder and a connection, and complete the page.

Packages on Integration Server localhost:5555

- Default
- MyConnections
- WmART
- WmAssetPublisher

Folder Name: MyConnections

Connection Name: MyConnectionsConnection

Listener Name: MyConnectionsListener

Table of IDL Programs and their related IS Services:

RPC Program Name	IS Service Name
ADD	EXAMPLE:ADD
MULTIPLY	EXAMPLE:MULTIPLY
SUBTRACT	EXAMPLE:SUBTRACT

RPC Listener Connection to EntireX

Broker ID: localhost:1971

Server Address: RPC/SRV1/CALLNAT

User ID:

Password:

Encoding:

? < Back Next > Finish Cancel

➤ To define a listener

- 1 Select an Integration Server package where the listener will be stored.
- 2 Specify the names for **Folder Name** (default: *library name*), **Connection Name** (default: *library nameConnection*) and **Listener Name** (default: *library nameListener*).

- 3 If necessary, edit the Broker settings for the Listener (**Broker ID, Server Address, User ID, Password, and Encoding**).
- 4 The check box **Overwrite existing Objects in Integration Server** can be used to re-generate the objects after you have changed the IDL file.



Notes:

1. The check box **Overwrite existing Objects in Integration Server** is useful for re-generating objects created previously. However, you cannot overwrite an RPC Listener Connection or a reliable RPC Listener Connection with a connection of a different type. If the connection is deleted with the Adapter Administration UI, it is not possible to overwrite the objects. In this case, you have to delete the adapter services in the Designer.
 2. When creating a connection, a package dependency is added such that the selected package depends on webMethods EntireX (the package `WmEntireX`) with the version currently used.
- 5 Choose **Finish**.

As a result, the following objects will be created:

- one connection of type EntireX RPC Listener, EntireX Reliable RPC Listener or EntireX Direct RPC Listener
- one listener object

For a connection of type **EntireX Reliable RPC Listener** the following objects will be created in addition for each IDL program:

- one notification object
- one trigger object
- one document type object

3

Mapping Integration Server Data Types to IDL

The signature of an Integration Server service specifies the data types for the parameters of the service. There are three data types: `String`, `Record` and `Object`. Parameters of type `String` are mapped to an IDL alphanumeric data type, parameters of type `Record` to IDL groups. Parameters of type `Object` cannot be mapped to an IDL data type and thus result in an error. If such an error occurs, the IDL program which contains the invalid data type mapping is written to the IDL file, however, all lines are comments.

Starting with version 9.7, a parameter of an EntireX Adapter service of type `Object` is mapped to an IDL binary data type. Depending on the target language, it is either mapped to `BV` (for `Natural` and `Other`) or to `BV256` (for `COBOL` and `PL/I`).

All parameters of type `String` are mapped to an IDL alphanumeric data type, available as variable (`AV`, `AVn`) or fixed (`An`) length. Which alphanumeric type is used is specified in the wizard page of the IDL Extractor for Integration Server or as a parameter of the generation service.

Parameters of type `String` may have an associated context type which is specified in the constraints of the parameter's properties. This content type influences the data type mapping. The content type was already considered in previous versions of the EntireX Adapter (version 9.6 or lower). However, its usage was not documented and the mapping was not yet complete. The handling of the content type was reworked with version 9.7. The following table shows both the new and the old mapping. The new mapping is different from the old one. If you want to use the old mapping with version 9.7 or higher, set the extended setting `watt.com.softwareag.entirex.wmadapter.extractor.contenttype.handling.pre97=true` (in the Integration Server administration page under **Settings > Extended**).

Integration Server Content Type	Software AG IDL Data Type			
	Adapter Version 9.7 or higher		Adapter Version 9.6 or lower	
boolean	L	Logical	L	Logical
decimal	Nx , $Nx.y$ or An , AV , AVn	Unpacked decimal or Alphanumeric ⁽¹⁾	$N10.2$	Unpacked decimal
float	F4	Floating point (small)	F4	Floating point (small)
double	F8	Floating point (large)	F8	Floating point (large)
Date	An , AV , AVn	Alphanumeric ⁽²⁾	D	Date
dateTime	An , AV , AVn	Alphanumeric ⁽²⁾	T	Time
base64Binary	An , AV , AVn	Alphanumeric ⁽²⁾	BV	Binary variable length
byte	I1	Integer (small)	I1	Integer (small)
unsignedByte	NU3	Unpacked decimal unsigned	I1	Integer (small)
short	I2	Integer (medium)	I2	Integer (medium)
unsignedShort	NU5	Unpacked decimal unsigned	I2	Integer (medium)
int	I4	Integer (large)	I4	Integer (large)
unsignedInt	NU10	Unpacked decimal unsigned	I4	Integer (large)
long	N19	Unpacked decimal	An , AV , AVn	Alphanumeric ⁽²⁾
unsignedLong	NU20	Unpacked decimal unsigned	An , AV , AVn	Alphanumeric ⁽²⁾
integer	N29	Unpacked decimal	An , AV , AVn	Alphanumeric ⁽²⁾
positiveInteger	NU29	Unpacked decimal unsigned	An , AV , AVn	Alphanumeric ⁽²⁾
nonPositiveInteger	N29	Unpacked decimal	An , AV , AVn	Alphanumeric ⁽²⁾
negativeInteger	N29	Unpacked decimal	An , AV , AVn	Alphanumeric ⁽²⁾
nonNegativeInteger	NU29	Unpacked decimal unsigned	An , AV , AVn	Alphanumeric ⁽²⁾
string	An , AV , AVn	Alphanumeric ⁽³⁾	An , AV , AVn	Alphanumeric ⁽²⁾
all others	An , AV , AVn	Alphanumeric ⁽²⁾	An , AV , AVn	Alphanumeric ⁽²⁾



Notes:

1. If the content type specifies a total number of digits and a number of fraction digits, then $Nx.y$ is used. If only a total number of digits is specified, then Nx is used. If no total number of digits is specified, or the total number exceeds the maximum allowed value, or the number of fraction digits exceeds the maximum allowed value, then a string according to ⁽²⁾ is used. The maximum allowed value for the total number of digits is 29, for COBOL it is 31. The maximum allowed value for the number of fraction digits is 7, for COBOL it is 31 and for Natural it is 29.

2. Depending on what is specified in the wizard page of the IDL Extractor for Integration Server or as a parameter of the generation service, the IDL data type AV , AV_n or A_n is used.
3. If the content type specifies a length len , then A_{len} is used. If the content type specifies a maximum length max , then AV_{max} or A_{max} is used, see ⁽²⁾ for the choice between A versus AV . If neither length nor maximum length is specified, the choice is the same as in ⁽²⁾.

4 Using the Service

pub.wmentirex.listener:generateIDLfromService

Service `pub.wmentirex.listener:generateIDLfromService` generates an IDL file for a given list of Integration Server services. For each service, a program in the IDL file is created. If the IDL file already exists, it is extended with the newly created program definitions. Duplicate definitions for the same program are possible. In this case, the IDL file has to be corrected manually. Note that this service creates/updates only the IDL file. Creating or updating a listener with the definitions of the IDL file has to be done with *webMethods Integration Server Wrapper* of the EntireX Workbench. The following parameters are relevant. Parameter direction can be In or Out:

Parameter	I/O	Description
<code>serviceName String</code>	I	The full name of the Integration Server service (e.g. <code>folder1.folder2:service</code>). Multiple services can be specified using a semicolon (;) as delimiter.
<code>fileName String</code>	I	The name of the IDL file (with extension ".idl"). You cannot specify a folder name. The file is created in the folder <code><IntegrationServer_instance>/packages/WmEntireX/resources</code> .
<code>libraryName String</code>	I	The library name to be used in the IDL file. If the IDL file already exists, this parameter is ignored.
<code>stringType String</code>	I	The IDL data type which is used for string data types. Possible values are AV, AVn, or An. This parameter is optional, default is AV.
<code>language String</code>	I	The target language for the RPC clients. Possible values are COBOL, Natural, PL/I, or Other.
<code>result String</code>	O	Either a success message if the generation was successful (which includes the full pathname of the IDL file) or an error message.

5 Integration Server Preferences

- Integration Server Connections 22
- Setting Integration Server Preferences 24

The Integration Server preferences are used to manage Integration Server connections. This chapter applies both to the Integration Server Wrapper and the IDL Extractor for Integration Server.

Integration Server Connections

The Integration Server connections are responsible for the HTTP/HTTPS communication to the Integration Server. They are used in the wizards described in *Using the Integration Server Wrapper* and *Using the IDL Extractor for Integration Server* and are managed in the Integration Server preferences.

An Integration Server connection contains the following information:

- Server name (required, consists of hostname and optional port number, where the default port number is 5555)
- User name
- Password
- optional parameters for SSL (HTTPS):
 - Truststore (name of the file)
 - Verify hostname
 - Optional parameters for client verification:
 - Keystore (name of the file)
 - Password for the Keystore

This information can be specified in the following dialog:

 **Notes:**

1. The only required field is **Server**. Enter the hostname of the Integration Server including an optional port number. If no port number is specified, port number defaults to "5555". The **Integration Server Authentication** can be passed with the **User** and **Password** fields.
2. Optional settings are for secure connections. The **Truststore for HTTPS** contains all signed certificates and must be a valid truststore.
3. The check box **Verify host name** checks that the hostname is entered in the stored certificate.
4. When the Integration Server has **Client Authentication** enabled, you can specify your **Keystore** file and keystore **Password**.
5. For managing Integration Server connections, see *Integration Server Preferences*.

