

webMethods Module for Flat File Built-In Services Reference

Version 10.7

January 2022

This document applies to webMethods FlatFile Module 10.7 and to all subsequent releases.

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

Copyright © 2007-2022 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 <https://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 <https://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 <https://softwareag.com/licenses/> and/or in the root installation directory of the licensed product(s).

Document ID: WFF-BIS-107-20220112

Table of Contents

About this Guide	5
Document Conventions.....	6
Online Information and Support.....	7
Data Protection.....	7
1 wmFlatFile Package	9
Flat File Folder.....	10

About this Guide

- Document Conventions 6
- Online Information and Support 7
- Data Protection 7

The *webMethods Module for EDI Built-In Services Reference* describes the built-in services provided with the webMethods Module for EDI (Module for EDI). The services that are documented in this guide are provided in the WmEDI and WmEDIforTN packages that are installed with the webMethods Module for EDI.

Services are also installed with a standard installation of webMethods Integration Server and webMethods Trading Networks (Trading Networks). You will find documentation for the built-in services provided with webMethods Integration Server in the *webMethods Integration Server Built-In Services Reference* for your release and documentation for services provided with Trading Networks in the *webMethods Trading Networks Built-In Services Reference* for your release.

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
Narrowfont	Identifies service names and locations in the format <i>folder.subfolder.service</i> , APIs, Java classes, methods, properties.
<i>Italic</i>	Identifies: Variables for which you must supply values specific to your own situation or environment. New terms the first time they occur in the text. References to other documentation sources.
Monospace font	Identifies: Text you must type in. Messages displayed by the system. Program code.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the symbol.
[]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information and Support

Software AG Documentation Website

You can find documentation on the Software AG Documentation website at <https://documentation.softwareag.com>.

Software AG Empower Product Support Website

If you do not yet have an account for Empower, send an email to empower@softwareag.com with your name, company, and company email address and request an account.

Once you have an account, you can open Support Incidents online via the eService section of Empower at <https://empower.softwareag.com/>.

You can find product information on the Software AG Empower Product Support website at <https://empower.softwareag.com>.

To submit feature/enhancement requests, get information about product availability, and download products, go to [Products](#).

To get information about fixes and to read early warnings, technical papers, and knowledge base articles, go to the [Knowledge Center](#).

If you have any questions, you can find a local or toll-free number for your country in our Global Support Contact Directory at https://empower.softwareag.com/public_directory.aspx and give us a call.

Software AG Tech Community

You can find documentation and other technical information on the Software AG Tech Community website at <https://techcommunity.softwareag.com>. You can:

- Access product documentation, if you have Tech Community credentials. If you do not, you will need to register and specify "Documentation" as an area of interest.
- Access articles, code samples, demos, and tutorials.
- Use the online discussion forums, moderated by Software AG professionals, to ask questions, discuss best practices, and learn how other customers are using Software AG technology.
- Link to external websites that discuss open standards and web technology.

Data Protection

Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

1 **wmFlatFile Package**

■ Flat File Folder	10
--------------------------	----

Flat File Folder

Use the elements in the Flat File folder to convert between Flat File documents and IS documents (IData objects), and to manage dictionary entries, entire flat file dictionaries, and flat file schemas.

Summary of Elements in the Flat File Folder

The following elements are available in this folder:

Element	Package and Description
pub.flatFile:convertToString	WmFlatFile. Converts an IS document (IData object) to a flat file document based on the flat file schema that you specify.
pub.flatFile:convertToValues	WmFlatFile. Converts a flat file document to an IS document (IData object) based on the input flat file schema.
pub.flatFile:FormatService	WmFlatFile. Service that formats the field String in a flat file schema or dictionary and ensures that the value of the String meets the format restrictions of the format service.
pub.flatFile:getSupportedEncodings	WmFlatFile. Returns a list of supported encodings. This service will only report webMethods encodings, not Java defaults. That is, if you do not have converters.jar installed, it returns null.
pub.flatFile.generate:createDocumentType	WmFlatFile. Creates an IS document type that defines the XML representation of a flat file schema.
pub.flatFile.generate:createFFDictionary	WmFlatFile. Creates an empty flat file dictionary. This service throws an exception if the flat file dictionary you want to create already exists when the service is invoked.
pub.flatFile.generate:deleteFFDictionary	WmFlatFile. Deletes a flat file dictionary.
pub.flatFile.generate:deleteFFDictionaryEntry	WmFlatFile. Deletes a single entry from a flat file dictionary.
pub.flatFile.generate:deleteFFSchema	WmFlatFile. Deletes a flat file schema.
pub.flatFile.generate:FFDictionary	WmFlatFile. This IS document type defines the format to use when supplying a flat file dictionary or dictionary entry (in the

Element	Package and Description
	<i>FFXML</i> variable) and the format that services return (in the <i>FFXML</i> variable) when you are retrieving a flat file dictionary or dictionary entry.
pub.flatFile.generate:FFSchema	WmFlatFile. This IS document type defines the format to use when supplying a flat file schema (in the <i>FFXML</i> variable) and the format that services return (in the <i>FFXML</i> variable) when you are retrieving a flat file schema.
pub.flatFile.generate:findDependants	WmFlatFile. Returns the names of all flat file schemas and dictionaries that are dependent on a given flat file dictionary.
pub.flatFile.generate:findReferences	WmFlatFile. Returns the names of all flat file dictionaries that are referenced by a given flat file dictionary or flat file schema.
pub.flatFile.generate:getFFDictionaryAsXML	WmFlatFile. Returns a dictionary as an XML string.
pub.flatFile.generate:getFFDictionaryEntryAsXML	WmFlatFile. Returns a single dictionary entry as an XML string.
pub.flatFile.generate:getFFSchemaAsXML	WmFlatFile. Returns the specified flat file schema as an XML string.
pub.flatFile.generate:listFFDictionaryEntries	WmFlatFile. Lists all entries in a specified flat file dictionary that are of a specified type.
pub.flatFile.generate:saveXMLAsFFDictionary	WmFlatFile. Creates a flat file dictionary in the Integration Server namespace by converting the specified flat file dictionary that is in XML format into a namespace flat file dictionary.
pub.flatFile.generate:saveXMLAsFFSchema	WmFlatFile. Creates a flat file schema in the Integration Server namespace by converting the specified flat file schema that is in XML format into a namespace flat file schema.
pub.flatFile.generate:updateFFDictionaryEntryFromXML	WmFlatFile. Updates one or more entries in a flat file dictionary in the Integration Server namespace.

pub.flatFile:convertToString

WmFlatFile. Converts an IS document (IData object) to a flat file document based on the flat file schema that you specify.

By default, this service returns the document as a string, but you can set a flag to optionally return the document as a byte array instead.

Note:

This service does not validate the document.

Input Variables

<i>ffValues</i>	Document. The IData object representing the flat file.
<i>ffSchema</i>	String. Namespace name of the flat file schema to use to convert the given IS document to a string.
<i>spacePad</i>	String. Optional. How to position the records in the flat file.

<u>Value</u>	<u>Description</u>
left	Left justify the records (add blank spaces to the right of the records) before the records are written to the output. This is the default.
right	Right justify the records (add blank spaces to the left of the records) before the records are written to the output.
none	No spaces added.

signalError **String.** Whether to create errors in the output.

<u>Value</u>	<u>Description</u>
false	Do not create errors in output.
true	Create errors in output.

If you are upgrading from webMethods Integration Server version 4.6, to enable left or right justification you must add the following line to the *Integration Server_directory* \instances*instance_name*\packages\WmFlatFile\config\ff file:

```
spacePadJustifies=false
```

Then, reload the WmFlatFile package so that this configuration setting will take effect. For details, see the *Flat File Schema Developer's Guide* or *webMethods Service Development Help*.

noEmptyTrailing Fields **String.** Whether trailing empty fields are to be removed from the output. Used only with records that have delimited fields.

<u>Value</u>	<u>Description</u>
--------------	--------------------

	<code>true</code>	Trailing empty fields will be removed from the output. For example, if it is set to true, the output for a record with empty trailing fields looks like the following: AAA*01*02! (where ! is used as segment terminator). This is the default.
	<code>false</code>	A field separator remains to denote an empty field. For example, if it is set to false, the output for a record with empty trailing fields looks like the following: AAA*01*02*****! (where ! is used as segment terminator).
<i>noEmptyTrailingSubFields</i>		String. Whether trailing empty subfields are to be removed from the output. Used only with records that have delimited fields. If no value is specified for the <i>noEmptyTrailingSubFields</i> parameter, Integration Server uses the value set for the <i>noEmptyTrailingFields</i> parameter.
	Value	Description
	<code>true</code>	Trailing empty subfields will be removed from the output.
	<code>false</code>	A field separator remains to denote an empty subfield.
<i>delimiters</i>		Document. Optional. The separator characters used to construct the output string. To specify a delimiter, you can specify: <ul style="list-style-type: none"> ■ One character or character representation (for example, *, \n for line terminator, \t for tab) ■ Hexadecimal value with prefix "0x" (for example, 0x09, 0x13) ■ Octal value with prefix "0" or decimal value (for example, 009, 013) ■ Unicode characters (for example, \u XXXX where XXXX represents the Unicode value of the character)
	Value	Description
	<i>record</i>	String. Character to use to separate records. If you want to specify the two-character carriage return line feed (CRLF) characters, specify \r\n.
	<i>field</i>	String. Character to use to separate fields.
	<i>subfield</i>	String. Character to use to separate subfields.
	<i>release</i>	String. Character to use to ignore a <i>record</i> , <i>field</i> , or <i>subfield</i> delimiter in a field. If a release character occurs in a field or subfield before the delimiter, it will be prefixed with <i>release</i> before being written to the output <i>string</i> .
	<i>quotedRelease</i>	String. Character to use to ignore a <i>record</i> , <i>field</i> , or <i>subfield</i> delimiter in a field. If a quoted release character occurs in a field or subfield before the delimiter, it will be prefixed with

quotedRelease before being written to the output *string*. The string is pre- and appended with the quoted release character.

For example, if * is a delimiter, the field value is a*b, and the quoted release character is ", the string appears as "a*b".

lineSeparator **String.** Optional. Character to use to override the input record delimiter when the following pre-condition is met and to handle the operating system specific newline.

The record delimiter must be either "\r\n" [CRLF] or "\r" [CR] or "\n" [LF] and the parser is of Delimiter Type.

Note:

Integration Server ignores *lineSeparator* if the above pre-condition is not met.

Value	Description
unix	Integration Server overrides the record delimiter with LF or "\n".
windows	Integration Server overrides the record delimiter with CRLF or "\r\n".
mac	Integration Server overrides the record delimiter with CR or "\r".
auto	Integration Server overrides the record delimiter with the operating system specific line separator.

FormatInfo **Document.** Any values mapped to the *FormatInfo* variable will be passed unmodified to all format services invoked by *convertToString* and *convertToValues*.

outputFileName **String.** Optional. If you want the output returned in a file instead of in the *string* output variable, provide the name of the file you want created as a result of this service.

Encoding **String** The type of encoding used to write data to the output file. The default encoding is UTF-8.

Note:

If the flat file document contains multi-byte encodings, you must install the Extended Character Set using the Software AG Installer. For more information about installing the Extended Character Set, see the

“Configuring the flat file byte count parser” section of the *Flat File Schema Developer’s Guide*.

sortInput

String. Optional. Whether you want the service to sort the input records to match the flat file schema specified in *ffSchema*. You should specify `true` for *sortInput* if the data in *ffValues* is not in the same order as defined by *ffSchema*.

Value	Description
<code>true</code>	<p>You want the service to sort the input records to match the flat file schema.</p> <p>If you select to sort the input records, note that:</p> <ul style="list-style-type: none"> ■ The service will run slower. ■ All undefined records will be sorted <i>after</i> the defined records. ■ The order of the undefined records appear in the final document is random. <p>If there are multiple records at the same level with the same name, the order they appear in the final document is random.</p>
<code>false</code>	You do not want the service to sort the input records to match the flat file schema. The input records must match the order of the flat file schema. This is the default.

returnAsBytes

Value	Description
<code>false</code>	Returns the document as a string. This is the default.
<code>true</code>	Returns the document as a byte array instead of a string. This setting is useful (but optional) when parsing multi-byte encodings.

Output Variables

string

String. Data that represents the flat file document.

bytes

Object. If the input variable *returnAsBytes*=`true`, returns the output as a byte array encoded using the specified encoding. The string value is not returned.

errorArray

Object String array containing messages pertaining to errors that occurred during conversion. If no errors are encountered, this contains a value of `null`.

Usage Note

When the `pub.flatFile:convertToString` service executes, the field that is defined to start after the end of the fixed length record will not be included in the output data if the following conditions are met:

- The flat file schema uses a fixed length record delimiter.
- The flat file schema contains a fixed position field that begins beyond the defined length of the fixed length record.
- The input to the `pub.flatFile:convertToString` service contains a value for the fixed position field that begins beyond the defined length of the fixed length record.

pub.flatFile:convertToValues

`WmFlatFile`. Converts a flat file document to an IS document (IData object) based on the input flat file schema.

Input Variables

<i>ffData</i>	Object. The flat file input with type of <code>String</code> , <code>InputStream</code> , or <code>ByteArray</code> .
<i>ffSchema</i>	String. The full name of the flat file schema object used to parse the <i>ffData</i> object.
<i>ffIterator</i>	Object. Optional. An object that encapsulates and keeps track of the input data during processing. It is used only when the <i>iterate</i> variable has been set to true.
<i>encoding</i>	String. Optional. The encoding of the <code>InputStream</code> passed in to <i>ffData</i> . The default encoding is UTF-8.

Note:

If the flat file document contains multi-byte encodings, you must install the Extended Character Set using the Software AG Installer. For more information about installing the Extended Character Set, see the "Configuring the flat file byte count parser" section of the *Flat File Schema Developer's Guide*.

<i>delimiters</i>	Document. Optional. An <code>IData</code> object that contains the segment terminator and the field and subfield separators. If the delimiter is <code>null</code> , it will be located using the information defined in the flat file schema. To specify a delimiter, you can specify:
-------------------	--

- One character or character representation (for example, `*`, `\n` for line terminator, `\t` for tab)
- Hexadecimal value with prefix "0X" (for example, `0X09`, `0X13`)
- Octal value with prefix "0" or decimal value (for example, `011`, `023`)
- Unicode characters (for example, `\u XXXX` where `XXXX` represents the Unicode value of the character)

- The space character.

Important:

If you specify one delimiter value, you must specify *all* values. Specifying one of these values will override any information set in the flat file schema.

Variable	Description
<i>record</i>	String. Character used to separate records. If you want to specify the two-character carriage return line feed (CRLF) characters, specify <code>\r\n</code> .
<i>field</i>	String. Character used to separate fields.
<i>subfield</i>	String. Character used to separate subfields.
<i>release</i>	String. Character used to ignore a <i>record</i> , <i>field</i> , or <i>subfield</i> delimiter in a field. If a release character occurs in a field or subfield before the delimiter, it will be prefixed with the <i>release</i> before being written to the output <i>ffValues</i> .
<i>quotedRelease</i>	String. Character to use to ignore a <i>record</i> , <i>field</i> , or <i>subfield</i> delimiter in a field. If a quoted release character occurs in a field or subfield before the delimiter, it will be prefixed with <i>quotedRelease</i> before being written to the output <i>string</i> . The string is pre- and appended with the quoted release character. For example, if * is a delimiter, the field value is a*b, and the quoted release character is ", the string appears as "a*b".
<i>FormatInfo</i>	Document. Any values mapped to the <i>FormatInfo</i> variable will be passed unmodified to all format services invoked by <code>convertToString</code> and <code>convertToValues</code> .
<i>iterate</i>	String. Optional. Whether you want to process the input all at one time.
<i>batchsize</i>	String Optional. Whether you want to process the specified number of records. The input value of this field is a positive integer. This field is applicable only when the <i>iterate</i> field value is set to true. If you do not specify this field value, default value is set to 1.
Value	Description
true	Processes top-level records (children of the document root) in the flat file schema one at a time. After all child records of the top-level record are processed, the iterator moves to the top-level of the next record in the flat file schema, until all records are processed.
false	Processes all input data at one time. This is the default.

createIfNull **String** Optional. Whether to create the IData object if all the fields are null.

Value	Description
true	No IS document (IData object) will be created if all the fields are null. This is the default.
false	Always create IS document even though all the fields are null.

skipWhiteSpace **String**. Optional. Whether white space at the beginning of records will be ignored.

Note:
The fixed length record parser ignores *skipWhiteSpace*; it preserves white space.

Value	Description
true	Ignore white spaces at the beginning of a record. This is the default.
false	Record is used as it is identified (useful for positional data record).

keepResults **String**. Optional. Whether to return the parsed data in the *ffValues* output parameter.

Value	Description
true	The parsed <i>ffData</i> will be returned in the output <i>ffValues</i> . This is the default.
false	<i>ffValues</i> will not return data. Use this option when validating the structure of the <i>ffData</i> against the given flat file schema.

validate **String**. Optional. Whether to return error messages that describe how *ffData* differs from the flat file schema.

Value	Description
true	Do not return error messages describing how <i>ffData</i> differs from the specified flat file schema. This is the default.
false	Return errors describing how the given <i>ffData</i> violates the constraints described in the flat file schema.

returnErrors **String**. Optional. Whether to return the validation errors. Validation errors are returned only if *validate* is set to true.

Value	Description
--------------	--------------------

<code>asArray</code>	Return any validation errors with the <i>ffData</i> in an array called <i>errors</i> . This is the default.
<code>inResults</code>	Return validation errors in the <i>ffValues</i> object.
<code>both</code>	Return validation errors in both <i>errors</i> and <i>ffValues</i> .
<i>maxErrors</i>	String. Optional. The maximum number of errors that can be returned from one record. When the flat file parser encounters more than the maximum number of errors within a record, the parser will stop parsing and return the parsed data and errors processed up until that point. Validation errors are returned only if <i>validate</i> is set to <code>true</code> .

flags **String.** Optional. Flags that you can set to govern `convertToValues` options.

Variable	Description
----------	-------------

<i>addRecordCount</i>	String. Whether you want the service to add an additional field (<i>@record-count</i>) to each parsed record in the resulting <i>IData</i> object (<i>ffValues</i>). The <i>@record-count</i> field is used to identify the record number of each parsed record.
-----------------------	---

Value	Description
-------	-------------

<code>true</code>	The <i>@record-count</i> field is added to each parsed record. This field contains the number of the parsed record. The first parsed record is 1, the second is 2, etc. If there are records that are undefined data, the count of the next defined record will reflect the undefined data. For example, if the <i>@record-count</i> field for a record is 2 and that record contains 5 undefined records, the <i>@record-count</i> field for the next defined record will be 8.
-------------------	--

<code>false</code>	The <i>@record-count</i> field is not added to each parsed record. This is the default.
--------------------	---

<i>detailedErrors</i>	String. Whether you want detailed conditional validation error information. This flag is only used when <i>validate</i> is <code>true</code> .
-----------------------	---

Value	Description
-------	-------------

<code>true</code>	When a conditional validation error occurs, the output <i>errors</i> variable will contain detail information about all the conditions that were violated. For more information, see <i>Flat File Schema Developer's Guide</i> .
-------------------	--

	false	When a conditional validation error occurs, the service does <i>not</i> provide detail error information. Conditional validators report only whether a condition failed validation with no additional information about the conditions that were violated. This is the default.
<i>skipToFirstRecord</i>		String. Whether you want the service to wait until it finds the first valid record before reporting invalid records as errors.
	Value	Description
	true	The service will wait until it finds the first valid record before reporting invalid records as errors. This is the default.
	false	The service will report invalid records as errors prior to locating the first valid record.
<i>trimWhitespace</i>		String. Whether you want the service to delete any blank spaces at the beginning of fields, at the end of fields, or both.
	Value	Description
	none	The service will not delete any blank spaces from fields. This is the default.
	left	The service will delete all blank spaces at the beginning of all fields.
	right	The service will delete all blank spaces at the end of all fields.
	both	The service will delete all blank spaces at the beginning and end of all fields.
<i>resultAsArray</i>		String. Whether you want the service to return the <i>ffValues</i> output parameter as an <code>IData[]</code> that can be mapped to the document types generated from the schema. An <code>IData[]</code> is a document List. The <i>resultAsArray</i> parameter is used only when the <i>iterate</i> input parameter is set to true.
	Value	Description
	false	The service returns the <i>ffValues</i> output parameter as an <code>IData[]</code> that can be mapped to the document types generated from the schema. This is the default.

`true` The service returns the *ffValues* output parameter as an `IData` object and not as an `IData[]`.

Output Variables

ffValues **Document.** The `IData` object that represents the input flat file data.

ffIterator **Object.** Optional. An object that encapsulates and keeps track of the input records during processing. It is used only when the *iterate* variable has been set to `true`. When all input data has been processed, the object becomes `null`. When the *ffIterator* variable is `null`, you should exit the LOOP to discontinue processing.

isValid **String** Whether flat file contains validation errors.

Value	Description
<code>true</code>	The <i>validate</i> input variable was set to <code>true</code> and no errors were found.
<code>false</code>	The <i>validate</i> input variable was set to <code>true</code> and errors were found, or the <i>validate</i> input variable was set to <code>false</code> .

errors **String.** Optional. An array containing the validation errors, if any, that were found in *ffData*. For more information about validation error codes see *Flat File Schema Developer's Guide*.

Usage Note

If you specified a default record definition by which the `pub.flatFile:convertToValues` service parses the IS document (`IData` object), the service displays the resulting *recordWithNoID* document as a child of the document above it, in an array.

To display the *recordWithNoID* record as a child of the root, change the value of the `recWithNoIDLike46` to `true` in the *Integration Server_directory* \instances*instance_name*\packages\WmFlatFile\config\ff file and reload the WmFlatFile package so that this configuration setting will take effect. For more information, see the *Flat File Schema Developer's Guide*.

If some or all of the flat files that you process encounter problems because of memory constraints, set the *iterator* variable in the `pub.flatFile:convertToValues` service to `true` to process top-level records (children of the document root) in the flat file schema one at a time. After all child records of the top-level record are parsed, the `pub.flatFile:convertToValues` service returns and the iterator moves to the top-level of the next record in the schema, until all records are parsed. This parsing should be done in a flow service using a REPEAT step where each time the `pub.flatFile:convertToValues` service returns, the results are mapped and dropped from the pipeline to conserve memory. If the results were kept in the pipeline, out-of-memory errors might occur.

pub.flatFile:FormatService

WmFlatFile. Service that formats the field String in a flat file schema or dictionary and ensures that the value of the String meets the format restrictions of the format service.

Use this specification when you create format services for fields in a flat file schema or dictionary. The format service is invoked for a field when the [pub.flatFile:convertToValues](#) and [pub.flatFile:convertToString](#) services are invoked. You create a format service to format the field String and ensure that the value of the String meets the format restrictions of the format service. When creating a particular format service for use with the **Format Service** property in a flat file schema or dictionary, the service you select must implement the `pub.flatFile:FormatService` specification (located on its **Input/Output** tab).

Important:

If a particular field does not have a value (that is, a value is not returned in the IS document (IData object) for the [pub.flatFile:convertToValues](#) service or is not present in the input data for the [pub.flatFile:convertToValues](#) service) the format service assigned to that field will not be executed.

Input Variables

<i>value</i>	String. The field value to format.						
<i>direction</i>	String. Indicates the type of formatting to be applied to the field. Specify one of the following:						
	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>convertToString</td> <td>This field is in an outbound document and needs its internal format converted to its external format.</td> </tr> <tr> <td>convertToValues</td> <td>This field is in an inbound document and needs its external format converted to its internal format.</td> </tr> </tbody> </table>	Value	Description	convertToString	This field is in an outbound document and needs its internal format converted to its external format.	convertToValues	This field is in an inbound document and needs its external format converted to its internal format.
Value	Description						
convertToString	This field is in an outbound document and needs its internal format converted to its external format.						
convertToValues	This field is in an inbound document and needs its external format converted to its internal format.						
<i>validate</i>	String. The value of the input parameter <i>validate</i> from the pub.flatFile:convertToValues service.						
	<table> <tr> <td>true</td> <td>The value of the <code>convertToValues</code> <i>validate</i> parameter is true (validate).</td> </tr> <tr> <td>false</td> <td>The value of the <code>convertToValues</code> <i>validate</i> parameter is false (do not validate). This value is always false when the value of the <i>direction</i> parameter is <code>convertToString</code>.</td> </tr> </table>	true	The value of the <code>convertToValues</code> <i>validate</i> parameter is true (validate).	false	The value of the <code>convertToValues</code> <i>validate</i> parameter is false (do not validate). This value is always false when the value of the <i>direction</i> parameter is <code>convertToString</code> .		
true	The value of the <code>convertToValues</code> <i>validate</i> parameter is true (validate).						
false	The value of the <code>convertToValues</code> <i>validate</i> parameter is false (do not validate). This value is always false when the value of the <i>direction</i> parameter is <code>convertToString</code> .						
<i>minLength</i>	String. Enables you to validate the minimum length of a field. If the field is extracted via a Fixed Position Extractor, this is the number of bytes that are extracted. If the field is not extracted via the Fixed Position Extractor and a Length Validator is associated with this field, this is the minimum length that will be considered valid. Otherwise, this parameter will not be present in the pipeline.						

maxLength **String.** Enables you to validate the maximum length of a field. If the field is extracted via a Fixed Position Extractor, this is the number of bytes that are extracted. If the field is not extracted via the Fixed Position Extractor and a Length Validator is associated with this field, this is the maximum length that will be considered valid. If the maximum length is unlimited (-1) or there is no Length Validator, this parameter will not be present in the pipeline.

FormatInfo **Document.** Information that can be used by individual formatting services. This information can be obtained from one of 3 locations:

- `convertToString` – You can specify *FormatInfo* in addition to the delimiter information for a call to this service.
- `convertToValues` – If delimiter information is explicitly passed into the `convertToValues` service, *FormatInfo* can be specified.
- From the UNEDIFACT UNA segment – The EDI document type automatically extracts the decimal separator from the UNA segment.

The only format services that use this feature are the decimal formatting services (for implied decimal and decimal formats). The *FormatInfo* IS document should contain a string called *DecimalCharacter*. If the decimal character is ',' the number would be formatted as 100,10 (European format) instead of 100.10, as is common in the US.

Note:

Changes to the data in this object will be reflected in all other format services that are invoked during execution of `convertToString` and `convertToValues`.

Output Variables

formattedValue **String.** The field value with appropriate formatting applied.

meetsFormat **String.** Whether the value could be formatted properly.

Value	Description
true	Indicates that the value could be properly formatted.
false	Indicates that the value could not be properly formatted.

errorMessage **String.** If *meetsFormat* is false, this parameter provides a text message describing the formatting error.

valueToValidate **String.** The value that will be used by the validator for this field. If this value is not present, the value passed in the input variable *value* will be validated. This field is used only for situations in which the input variable *validate* is set to true.

maxLength **String.** Enables you to validate the maximum length of a field. If the field is extracted via a Fixed Position Extractor, this is the number of bytes that are extracted. If the field is not extracted via the Fixed Position Extractor and a Length Validator is associated with this field, this is the maximum length that will be considered valid. If the maximum length is unlimited (-1) or there is no Length Validator, this parameter will not be present in the pipeline.

FormatInfo **Document.** Information that can be used by individual formatting services. This information can be obtained from one of 3 locations:

- `convertToString` – You can specify *FormatInfo* in addition to the delimiter information for a call to this service.
- `convertToValues` – If delimiter information is explicitly passed into the `convertToValues` service, *FormatInfo* can be specified.
- From the UNEDIFACT UNA segment – The EDI document type automatically extracts the decimal separator from the UNA segment.

The only format services that use this feature are the decimal formatting services (for implied decimal and decimal formats). The *FormatInfo* IS document should contain a string called *DecimalCharacter*. If the decimal character is ',' the number would be formatted as 100,10 (European format) instead of 100.10, as is common in the US.

Note:

Changes to the data in this object will be reflected in all other format services that are invoked during execution of `convertToString` and `convertToValues`.

pub.flatFile:getSupportedEncodings

WmFlatFile. Returns a list of supported encodings.

This service reports webMethods encodings only, not Java defaults. If the `converters.jar` file is not installed, the service returns null. To install the `converters.jar` file, in the Software AG Installer select the **Custom Character Encoding Support** option under **Integration Server and Microservices Runtime Libraries**.

Input Variables

None.

Output Variables

encodings **String List** A list of supported encodings.

pub.flatFile.generate:createDocumentType

WmFlatFile. Creates an IS document type that defines the XML representation of a flat file schema.

Input Variables

<i>FlatFileSchema</i>	String. The fully-qualified name of the flat file schema for which you want to generate an IS document type.
<i>PackageName</i>	String. The name of the Integration Server package in which you want the created IS document type to be placed.
<i>DocumentType Name</i>	String. The fully-qualified name that you want to assign to the created IS document type.

Output Variables

None.

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

sample.flatFile.generateFFSchema:delimited

sample.flatFile.generateFFSchema:fixedLength

pub.flatFile.generate:createFFDictionary

WmFlatFile. Creates an empty flat file dictionary. This service throws an exception is if the flat file dictionary you want to create already exists when the service is invoked.

Input Variables

<i>FFDictionaryName</i>	String. The fully-qualified name of the flat file dictionary you want to create.
<i>PackageName</i>	String. The name of the Integration Server package in which you want the created flat file dictionary to be placed.

Output Variables

None.

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

sample.flatFile.generateFFSchema:delimited

sample.flatFile.generateFFSchema:fixedLength

pub.flatFile.generate:deleteFFDictionary

WmFlatFile. Deletes a flat file dictionary.

Before deleting the dictionary, the Integration Server determines if other dictionaries depend on the dictionary being deleted, and gives the user the option of canceling the deletion.

Input Variables

FFDictionaryName **String.** The fully qualified name of the flat file dictionary that you want to delete.

Output Variables

deleted **String.** Whether the flat file dictionary was successfully deleted; *deleted* will be either `true` or `false`.

Value	Description
<code>true</code>	The flat file dictionary was successfully deleted.
<code>false</code>	The flat file dictionary was <i>not</i> successfully deleted.

Usage Note

Before you run this service, you should run the [pub.flatFile.generate:findDependants](#) service to return the names of all flat file schemas and dictionaries that are dependent on the dictionary you are deleting.

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

sample.flatFile.generateFFSchema:delimited

sample.flatFile.generateFFSchema:fixedLength

pub.flatFile.generate:deleteFFDictionaryEntry

WmFlatFile. Deletes a single entry from a flat file dictionary.

Input Variables

<i>FFDictionaryName</i>	String. The fully-qualified name of the flat file dictionary that contains the entry that you want to delete.
<i>EntryName</i>	String. The name of the entry that you want to delete.
<i>EntryType</i>	String. The type of entry that you are deleting. Specify Record, Composite, or Field.

Output Variables

deleted **String.** Whether the flat file dictionary entry was successfully deleted; *deleted* will be either true or false.

Value	Description
true	The flat file dictionary entry was successfully deleted.
false	The flat file dictionary entry was <i>not</i> successfully deleted.

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

sample.flatFile.generateFFSchema:delimited

sample.flatFile.generateFFSchema:fixedLength

pub.flatFile.generate:deleteFFSchema

WmFlatFile. Deletes a flat file schema.

Input Variables

<i>FFSchemaName</i>	String. The fully-qualified name of the flat file schema that you want to delete.
---------------------	--

Output Variables

deleted **String.** Whether the flat file schema was successfully deleted; *deleted* will be either `true` or `false`.

<u>Value</u>	<u>Description</u>
<code>true</code>	The flat file schema was successfully deleted.
<code>false</code>	The flat file schema was <i>not</i> successfully deleted.

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

sample.flatFile.generateFFSchema:delimited

sample.flatFile.generateFFSchema:fixedLength

pub.flatFile.generate:FFDictionary

WmFlatFile. This IS document type defines the format to use when supplying a flat file dictionary or dictionary entry (in the FFXML variable) and the format that services return (in the FFXML variable) when you are retrieving a flat file dictionary or dictionary entry.

The structure for this IS document type is defined in the following XML schema:

Integration Server_directory \instances*instance_name*\packages\WmFlatFile\pub\FFGeneration.xsd

Variables

FFDictionary **Document.** The dictionary entries that you want to add or update. *FFDictionary* has the following structure:

RecordDictionary
Entry **Document List.** Optional. The dictionary entries for records that you want to add or update in the flat file dictionary. Leave this null if you do not want to add or update record entries.

<u>Variable</u>	<u>Description</u>
<i>EntryName</i>	String. The name of the record.
<i>RecordDefinition</i>	Document. The definition of the record. The information you specify in a record definition is the same as the information that you specify

when creating a flat file dictionary using the Flat File Schema Editor. For descriptions of the fields, see *webMethods Service Development Help*.

*Composite
DictionaryEntry*

Document List Optional. The dictionary entries for composites that you want to add or update in the flat file dictionary. Leave this null if you do not want to add or update composite entries.

Variable	Description
<i>EntryName</i>	String. The name of the composite.
<i>CompositeDefinition</i>	Document. The definition of the composite. The information you specify in a composite definition is the same as the information that you specify when creating a flat file dictionary using the Flat File Schema Editor. For descriptions of the fields, see <i>webMethods Service Development Help</i> .

FieldDictionary Entry **Document List.** Optional. The dictionary entries for fields that you want to add or update in the flat file dictionary. Leave this null if you do not want to add or update field entries.

Variable	Description
<i>EntryName</i>	String. The name of the field.
<i>FieldDefinition</i>	Document. The definition of the field. The information you specify in a field definition is the same as the information that you specify when creating a flat file dictionary using the Flat File Schema Editor. For descriptions of the fields, see <i>webMethods Service Development Help</i> .

Usage Notes

If you are using this IS document type to supply a flat file dictionary as input to the [pub.flatFile.generate:saveXMLAsFFDictionary](#), be sure to supply *all* dictionary entries. If you are using this IS document type to update an existing dictionary, provide only the entries that you want to add or update and invoke the [pub.flatFile.generate:updateFFDictionaryEntryFromXML](#) to update the flat file dictionary.

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

sample.flatFile.generateFFSchema:delimited

sample.flatFile.generateFFSchema:fixedLength

pub.flatFile.generate:FFSchema

WmFlatFile. This IS document type defines the format to use when supplying a flat file schema (in the *FFXML* variable) and the format that services return (in the *FFXML* variable) when you are retrieving a flat file schema.

The structure for this IS document type is defined in the following XML schema:

Integration Server_directory \instances*instance_name* \packages\WmFlatFile\pub\FFGeneration.xsd

Variables

FFSchema **Document.** The flat file schema that you want to add or update. *FFSchema* has the following structure:

<u>Variable</u>	<u>Description</u>
<i>Delimiters</i>	Document. The delimiters used in the flat files that adhere to this flat file schema. The information that you specify for <i>Delimiters</i> corresponds to the data you specify on the Flat File Definition tab in the Flat File Schema Editor. For a description of the fields, see <i>webMethods Service Development Help</i> .
<i>Document Structure</i>	Document. The structure of the flat files that adhere to this flat file schema. The information that you specify for <i>DocumentStructure</i> corresponds to the data you specify on the Flat File Structure tab in the Flat File Schema Editor. For a description of the fields, see <i>webMethods Service Development Help</i> .
<u>Variable</u>	<u>Description</u>
<i>Ordered</i>	String Whether the child records appear in the flat file in the order they are defined in the flat file schema.
<i>RecordStructure</i>	Document List Definitions of the records within the flat file.
<u>Variable</u>	<u>Description</u>

	<i>Ordered</i>	String. Whether the child records appear in the flat file in the order they are defined in the flat file schema.
	<i>RecordUsage</i>	Document. Information about how the record is used, including either the dictionary reference for this record or the definition of the record.
	<i>RecordStructure</i>	Document List. Child records of this record. This is a recursive reference to the <i>RecordStructure</i> defined in <i>FFSchema/DocumentStructure</i> .
<i>RecordParser</i>		Document. The type of record parser. In this IS document, specify only the one variable that corresponds to the type of record parser to use. That is, specify one of <i>FixedLengthParser</i> , <i>DelimitedParser</i> , <i>VariableLengthParser</i> , or <i>EDIParser</i> . For <i>DelimitedParser</i> , <i>VariableLengthParser</i> , and <i>EDIParser</i> , you do not need to specify a value; just have the variable in the pipeline.
<i>DefaultRecord Reference</i>		Document. Optional. The dictionary name and entry name that identifies the default record for the flat file schema. If you specify a default record, when using the flat file schema to parse a flat file schema, the default record is used for any record that cannot be recognized.
<i>Record Identifier</i>		Document. Where to locate the identifier to use to correlate a record in the flat file to a record definition in the flat file schema. Specify either the <i>NthFieldIdentifier</i> variable or the <i>FixedPositionIdentifier</i> variable: <ul style="list-style-type: none"> ■ Use <i>NthFieldIdentifier</i> to identify the field in the record (counting from zero) that contains the identifier. ■ Use <i>FixedPositionIdentifier</i> to identify the character position in the record (counting from zero) where the record identifier is located.

<i>UndefinedData Allowed</i>	<p>String. Whether you want the <code>pub.flatFile:convertToValues</code> service to generate undefined data errors when you use this flat file schema to convert a flat file to an IData object.</p> <ul style="list-style-type: none">■ Specify <code>true</code> if you want to allow undefined data and do not want the <code>pub.flatFile:convertToValues</code> service to flag undefined data errors.■ Specify <code>false</code> if you do not want to allow undefined data and you do want the <code>pub.flatFile:convertToValues</code> service to flag undefined data errors.
<i>Document Areas</i>	<p>String List. Areas for this flat file schema. An area is a way to associate an arbitrary string with a given record.</p>
<i>FloatingRecord</i>	<p>String. Optional. The name of the record that is defined in the schema as a floating record.</p> <div style="background-color: #f0f0f0; padding: 5px;"><p>Note: If the floating record has an alternate name, specify the alternate name.</p></div>
<i>Description</i>	<p>String. Description of the flat file schema.</p>

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

`sample.flatFile.generateFFSchema:delimited`

`sample.flatFile.generateFFSchema:fixedLength`

pub.flatFile.generate:findDependants

WmFlatFile. Returns the names of all flat file schemas and dictionaries that are dependent on a given flat file dictionary.

Input Variables

ffDictionaryName **String.** The name of the flat file dictionary whose dependents you want to find.

Output Variables

dependants **Document List** The dependent objects and the packages that contain them.

Variable	Description
<i>packageName</i>	String. The name of the package that contains the dependent object.
<i>name</i>	String. The name of the dependent object.

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

sample.flatFile.generateFFSchema:delimited

sample.flatFile.generateFFSchema:fixedLength

pub.flatFile.generate:findReferences

WmFlatFile. Returns the names of all flat file dictionaries that are referenced by a given flat file dictionary or flat file schema.

Input Variables

<i>name</i>	String. The name of the flat file dictionary or flat file schema whose references you want to find.
-------------	--

Output Variables

<i>references</i>	Document List The referenced objects and the packages that contain them.
-------------------	---

Variable	Description
<i>packageName</i>	String. The name of the package that contains the referenced object.
<i>name</i>	String. The name of the referenced object.

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

sample.flatFile.generateFFSchema:delimited

sample.flatFile.generateFFSchema:fixedLength

pub.flatFile.generate:getFFDictionaryAsXML

WmFlatFile. Returns a dictionary as an XML string.

Input Variables

FFDictionaryName **String.** The fully-qualified name of the flat file dictionary that you want returned as XML.

Output Variables

FFXML **String.** The returned flat file dictionary as an XML string. The returned XML string conforms to the [pub.flatFile.generate:FFDictionary](#) IS document type.

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

sample.flatFile.generateFFSchema:delimited

sample.flatFile.generateFFSchema:fixedLength

pub.flatFile.generate:getFFDictionaryEntryAsXML

WmFlatFile. Returns a single dictionary entry as an XML string.

Input Variables

FFDictionaryName **String.** The fully-qualified name of the flat file dictionary that contains the entry that you want returned as XML.

EntryName **String.** The name of the entry that you want to returned as XML.

EntryType **String.** The type of entry that you want returned. Specify Record, Composite, or Field.

Output Variables

FFXML **String.** The returned flat file dictionary entry as an XML string. The returned XML string conforms to the [pub.flatFile.generate:FFDictionary](#) IS document type.

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

sample.flatFile.generateFFSchema:delimited

sample.flatFile.generateFFSchema:fixedLength

pub.flatFile.generate:getFFSchemaAsXML

WmFlatFile. Returns the specified flat file schema as an XML string.

Input Variables

FFSchemaName **String.** The fully-qualified name of the flat file schema that you want returned as XML.

Output Variables

FFXML **String.** The returned flat file schema as an XML string. The returned XML string conforms to the [pub.flatFile.generate:FFDictionary](#) IS document type.

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

sample.flatFile.generateFFSchema:delimited

sample.flatFile.generateFFSchema:fixedLength

pub.flatFile.generate:listFFDictionaryEntries

WmFlatFile. Lists all entries in a specified flat file dictionary that are of a specified type.

Input Variables

FFDictionaryName **String.** The fully-qualified name of the flat file dictionary that contains the entries that you want listed.

EntryType **String.** The type of entries that you want listed. Specify `Record`, `Composite`, or `Field`.

Output Variables

EntryName **String List.** The list of returned flat file dictionary entries.

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

sample.flatFile.generateFFSchema:delimited

sample.flatFile.generateFFSchema:fixedLength

pub.flatFile.generate:saveXMLAsFFDictionary

WmFlatFile. Creates a flat file dictionary in the Integration Server namespace by converting the specified flat file dictionary that is in XML format into a namespace flat file dictionary.

If a flat file dictionary with the same name already exists in the Integration Server namespace, use the [pub.flatFile.generate:deleteFFDictionary](#) service to delete the flat file dictionary before invoking this service. This service throws an exception if a flat file dictionary with the same name already exists when it is invoked.

Input Variables

FFDictionaryName **String.** The fully-qualified name of the flat file dictionary that you want to create in the Integration Server namespace.

PackageName **String.** The name of the Integration Server package in which to save the flat file dictionary.

FFXML **String.** The flat file dictionary (as an XML string) that you want to create in the Integration Server namespace. The XML string must conform to the [pub.flatFile.generate:FFDictionary](#) IS document type.

Merk:

To see examples of how to supply the XML string in *FFXML* by mapping data from another file, see the samples provided in the WmFlatFileSamples package. For sample code that shows how to retrieve the data for *FFXML* from an XML file in the local file system, see *Flat File Schema Developer's Guide*.

maxNumOfErrors **String.** Optional. The maximum number of errors that you want returned. The default is 100.

The service ensures the flat file dictionary is valid before saving it in the Integration Server namespace. The validation occurs in two stages.

1. Structural validation of the XML.
2. Logical validation of the XML contents.

If structural validation errors occur, the service reports the structural validation errors, but does not proceed with logical validation. When the XML string contains no structural validation errors, the service proceeds with logical validation and reports any logical validation errors.

Output Variables

<i>saved</i>	String. Whether the flat file dictionary was saved successfully. It will have one of the following values.						
	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>true</td> <td>The flat file dictionary was successfully saved.</td> </tr> <tr> <td>false</td> <td>The flat file dictionary was <i>not</i> successfully saved.</td> </tr> </tbody> </table>	Value	Description	true	The flat file dictionary was successfully saved.	false	The flat file dictionary was <i>not</i> successfully saved.
Value	Description						
true	The flat file dictionary was successfully saved.						
false	The flat file dictionary was <i>not</i> successfully saved.						
<i>Errors</i>	String List. Optional. Errors that occurred while attempting to save the flat file dictionary to the Integration Server namespace.						
<i>Warnings</i>	String List. Optional. Warnings about the flat file dictionary that was created.						

Usage Note

Use this service to add a new flat file dictionary. Use the [pub.flatFile.generate:updateFFDictionaryEntryFromXML](#) if you want to update one or more entries in a flat file dictionary rather than creating a new flat file dictionary.

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

```
sample.flatFile.generateFFSchema:delimited
```

```
sample.flatFile.generateFFSchema:fixedLength
```

pub.flatFile.generate:saveXMLAsFFSchema

WmFlatFile. Creates a flat file schema in the Integration Server namespace by converting the specified flat file schema that is in XML format into a namespace flat file schema.

If a flat file schema with the same name already exists in the Integration Server namespace, use the [pub.flatFile.generate:deleteFFSchema](#) service to delete the flat file schema *before* invoking this service.

This service throws an exception if a flat file schema with the same name already exists when it is invoked.

Input Variables

FFSchemaName **String.** The fully-qualified name of the flat file schema that you want to create in the Integration Server namespace.

PackageName **String.** The name of the Integration Server package in which to save the flat file schema.

FFXML **String.** The flat file schema (as an XML string) that you want to create in the Integration Server namespace. The XML string must conform to the [pub.flatFile.generate:FFDictionary](#) IS document type.

Merk:

To see examples of how to supply the XML string in *FFXML* by mapping data from another file, see the samples provided in the *WmFlatFileSamples* package. For sample code that shows how to retrieve the data for *FFXML* from an XML file in the local file system, see *Flat File Schema Developer's Guide*.

maxNumOfErrors **String.** Optional. The maximum number of errors that you want returned. The default is 100.

The service ensures the flat file schema is valid before saving it in the Integration Server namespace. The validation occurs in two stages.

1. Structural validation of the XML.
2. Logical validation of the XML contents.

If structural validation errors occur, the service reports the structural validation errors, but does not proceed with logical validation. When the XML string contains no structural validation errors, the service proceeds with logical validation and reports any logical validation errors.

Output Variables

saved **String.** Whether the flat file schema was saved successfully. It will have one of the following values.

Value	Description
true	The flat file schema was successfully saved.
false	The flat file schema was <i>not</i> successfully saved.

Errors **String List.** Optional. Errors that occurred while attempting to save the flat file schema to the Integration Server namespace.

Warnings **String List.** Optional. Warnings about the flat file schema that was created.

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

sample.flatFile.generateFFSchema:delimited

sample.flatFile.generateFFSchema:fixedLength

pub.flatFile.generate:updateFFDictionaryEntryFromXML

WmFlatFile. Updates one or more entries in a flat file dictionary in the Integration Server namespace.

This service goes through all entries that you specify in the *FFXML* variable. If an entry with the same name and type already exists in the flat file dictionary, this service overwrites the existing entry. If the entry does *not* already exist, this service creates the entry in the specified flat file dictionary.

Input Variables

FFDictionaryName **String.** The fully-qualified name of the flat file dictionary that contains the entries that you are replacing, adding, or both.

FFXML **String.** The dictionary entries (as an XML string) that you want to use to replace an existing entry or that you want to add to the flat file dictionary. The XML string in *FFXML* must conform to the [pub.flatFile.generate:FFDictionary](#) IS document type.

Merk:

To see examples of how to supply the XML string in *FFXML* by mapping data from another file, see the samples provided in the WmFlatFileSamples package. For sample code that shows how to retrieve the data for *FFXML* from an XML file in the local file system, see *Flat File Schema Developer's Guide*.

maxNumOfErrors **String.** Optional. The maximum number of errors that you want returned. The default is 100.

The service ensures the flat file schema is valid before saving them in the flat file dictionary. The validation occurs in two stages.

1. Structural validation of the XML.
2. Logical validation of the XML contents.

If structural validation errors occur, the service reports the structural validation errors, but does not proceed with logical validation. When the XML string contains no structural validation errors, the service proceeds with logical validation and reports any logical validation errors.

Output Variables

saved **String.** Whether the dictionary entry was saved successfully. It will have one of the following values.

Value	Description
true	The dictionary entry was successfully saved.
false	The dictionary entry was <i>not</i> successfully saved.

Errors **String List.** Optional. Errors that occurred while attempting to save the entry to the flat file dictionary.

Warnings **String List.** Optional. Warnings about the dictionary entry that was updated or added.

Examples

For examples of how to use this service, see the following services in the certified samples area of the Knowledge Center on the Empower Product Support website at <https://empower.softwareag.com>.

sample.flatFile.generateFFSchema:delimited

sample.flatFile.generateFFSchema:fixedLength