WebSphere® Adapters
Version 7 Release 0

WebSphere Adapter for FTP User Guide Version 7 Release 0

WebSphere® Adapters
Version 7 Release 0

WebSphere Adapter for FTP User Guide Version 7 Release 0

IBM®

Note

Before using this information and the product it supports, read the information in "Notices" on page 229.

Contents

Chapter 1. Overview of WebSphere Adapter for FTP

With WebSphere Adapter for FTP, you can create integrated processes that use WebSphere Process Server and WebSphere Enterprise Service Bus to access files managed by an FTP server without having to know the details of FTP communications or protocols.

Once configured, the adapter acts like a service provider in a Service-oriented architecture (SOA) implementation, providing operations to send and retrieve files. The adapter is part of a module that is deployed to WebSphere Process Server or WebSphere Enterprise Service Bus.

The adapter exposes a service interface that hides the mechanics of how the data, or operations are obtained or run. Services outside of the module interact with the adapter instead of directly interacting with the FTP server, so authentication details (such as user name and password) that you provide when you set up a module are shielded from services outside of the module.

The module, which you create with the external service wizard in WebSphere Integration Developer, is a reusable unit designed to perform a specific inbound or outbound service. Each module uses a consistent interface and standard business objects, so applications consuming the service do not have to understand the lower-level details of the FTP server.

The following illustration shows how the adapter functions as part of an SOA implementation.

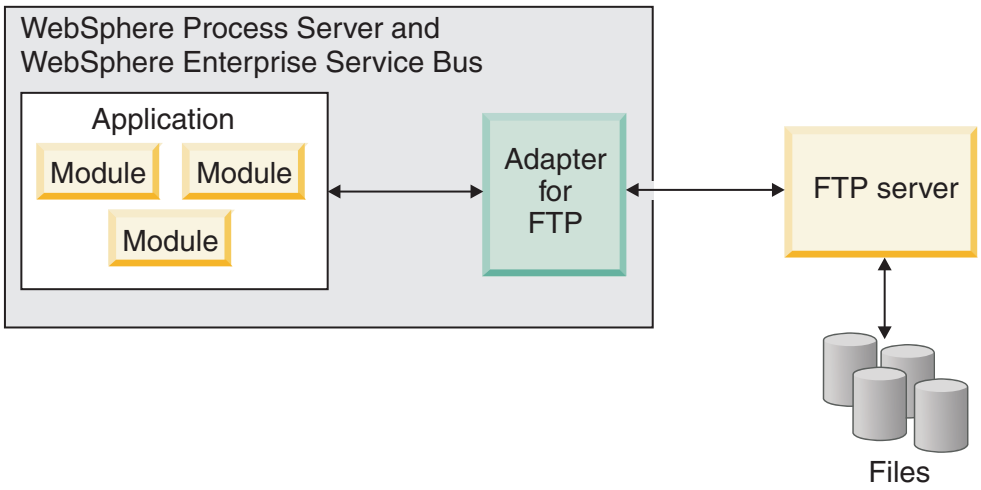


Figure 1. Adapter overview

What is new in this release

This version includes several new features that enhance the business flexibility, user experience, and performance of the adapter.

This information is also available at the WebSphere Adapters product support Web site, http://www.ibm.com/software/integration/wbiadapters/support/, which is periodically updated with the latest information.

WebSphere Adapter for FTP, version 7.0, includes the following new features:

- Support for Secure FTP in Explicit mode (FTP over SSL and FTP over TLS).
- Support for global elements (anonymous and named complex type) in business object definitions.
- Enhanced support for rules-editor, for setting file-filter-rules in the properties pane of the assembly editor in WebSphere Integration Developer, version 7.0.
- Migration
  - Migration of WebSphere Adapters version 6.x to WebSphere Adapters version 7.0 on IBM WebSphere Process Server.

Hardware and software requirements

The hardware and software requirements for WebSphere Adapters are provided on the IBM Support Web site.

To view hardware and software requirements for WebSphere Adapters, see http://www.ibm.com/support/docview.wss?uid=swg27006249

Additional information

The following links provide additional information you might need to configure and deploy your adapter:

- The compatibility matrix for WebSphere Business Integration Adapters and WebSphere Adapters identifies the supported versions of required software for your adapter. To view this document, go to the WebSphere Adapters support page and click **Compatibility Matrix** beneath the **Related** heading in the **Additional support links** section: http://www.ibm.com/software/integration/wbiadapters/support/.
- Technotes for WebSphere Adapters provide workaround and additional information that are not included in the product documentation. To view the technotes for your adapter, go to the following Web page, select the name of your adapter from the **Product category** list, and click the search icon: http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8&dc=DB520+D800+D900+DA900+DA800+DB560&dtm.

Technical overview of the Adapter for FTP

WebSphere Adapter for FTP provides the means for services running on WebSphere Process Server or WebSphere Enterprise Service Bus to communicate with one or more FTP servers.

The services are contained in a module, which consists of both a project in WebSphere Integration Developer and a unit of deployment to WebSphere Process Server. The module is packaged and deployed to WebSphere Process Server as an enterprise archive (EAR) file.

The module contains components, which are the actual services, imports and exports. Imports identify services outside of a module, making them callable from within the module. Exports allow components in a module to provide their services to external clients. Imports and exports require binding information, which specifies the means of transporting the data from the modules. The assembly editor in WebSphere Integration Developer sets up the imports and exports, lists the supported bindings, and simplifies their creation.

- An import is the point at which an SCA module accesses an external service (a service outside the SCA module) as if it were local. An import defines interactions between the SCA module and the service provider. An import has a binding and one or more interfaces.
- An export, also known as an endpoint, is an exposed interface from a Service Component Architecture (SCA) module that offers a business service to the outside world. An export has a binding that defines how the service can be accessed by service requesters, for example, the service requester may be a Web service.

Outbound processing

The Adapter for FTP supports outbound request processing. When the adapter receives a request, which is sent in the form of a business object from the module, it processes the request to perform operation on the files in the remote file system and returns the result, when applicable, in a business object.

The following illustration shows the outbound processing flow for the WebSphere Adapter for FTP.

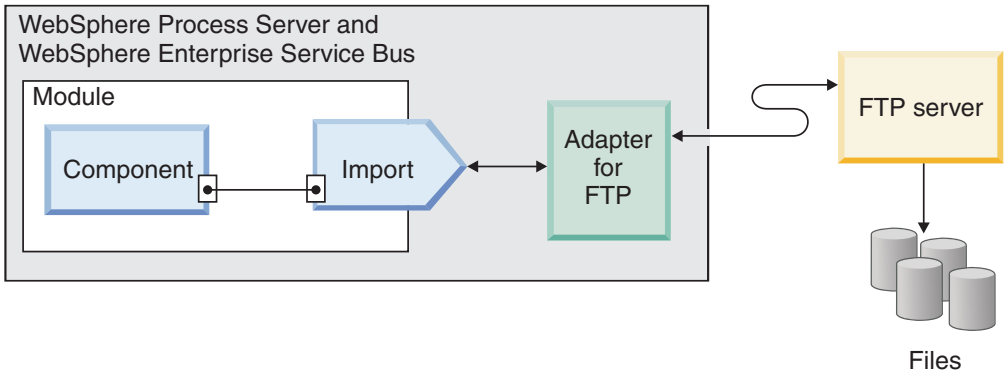


Figure 2. Outbound processing flow

Supported operations

An operation is an action that the adapter can perform on remote file systems accessible through an FTP server during outbound processing. The name of the operation typically indicates the type of action that the adapter takes, such as *Create* or *Append*.

During outbound processing, WebSphere Adapter for FTP supports the following operations.

Table 1. Supported outbound operations

Operation	Result
Create	The file with the specified name is created in the specified directory of the FTP server. The content of the file can either be sent as part of the request or it can be retrieved from the local file system. When the file content is received as part of the request, the adapter provides the option to archive the file on the adapter workstation before creating it. The file can be created in a staging directory and then sent to the actual directory. If a staging directory is not specified, the file is directly created in actual directory. After the file is created, the file name is sent back to the calling component to indicate that the file was created successfully. If the file to be created exists, a DuplicateRecord exception is sent, and the file is not created. The existing file is not overwritten. The adapter provides a feature to generate unique file names. See "Generating unique file names" on page 7. The adapter provides a feature to create a file sequence for the output files created. See "Generating a file sequence during Create operations" on page 6.
Append	The file with the specified name in the specified directory of the FTP server is appended with the content sent in the request. If the file to be appended exists, the content is appended, and the file name is sent back to the calling component indicating a successful response. If the staging directory is specified, the file to be appended is copied from the specified output directory to the staging directory, and the content is appended to that file in the staging directory. The appended file is then moved back to the original directory. If the file to be appended does not exist and the CreateIfFileNotExist property is set to `true`, the adapter creates a new file. If the file to be appended does not exist, a RecordNotFound exception is sent to the calling component.
Delete	The file in the specified directory is deleted on the FTP server and the adapter returns `true` to the calling component to indicate that the file was successfully deleted. If the file to be deleted does not exist, a RecordNotFound exception is sent to the calling component.

Table 1. Supported outbound operations (continued)

Operation	Result
Retrieve	The content of the file or files in the specified request is returned. The file content is split based on the SplittingFunctionClassName and SplitCriteria properties. The file content is transformed into a business object based on the configured data handler. After the content of the file is retrieved it is sent as the response. The file content can either be sent back to the calling component or saved to the local file system. If the file to be retrieved does not exist, a RecordNotFound exception is sent to the calling component. The adapter provides an option to delete the file from the FTP server directory after it is retrieved through the DeleteOnRetrieve property. The adapter supports an option to archive the file on the FTP server before it is deleted through the ArchiveDirectoryForDeleteOnRetrieve property. While configuring the Retrieve operation for data transformation, create custom retrieve wrappers like CustomerRetrieveWrapper or CustomerRetrieveWrapperBG, or OrderRetrieveWrapper or OrderRetrieveWrapperBG, and use the wrapper for the output type in the operation window. For a Retrieve operation without data transformation, the default wrapper RetrieveResponseWrapper is used. **Note:** The backward compatibility may use RetrieveResponseWrapper for retrieving XML data with data transformation.
Overwrite	This operation overwrites the file in the directory with the content specified in the request. After, the content is overwritten, the file name is sent back to the calling component indicating a successful response. The file to be overwritten is copied from the specified directory to the staging directory, if specified, and the content is overwritten for that file in the staging directory. The file is then moved back to the specified directory. If a staging directory is not specified, the content is overwritten on the file in the specified directory. If the file to be overwritten does not exist, and the CreateIfFileNotExist property is set to `true`, the adapter creates a new file. If the file to be overwritten does not exist, a RecordNotFound exception is sent to the calling component.
Exists	If the file name in the request exists in the specified directory or any of the sub folders, the adapter returns `true` and the full path of the file to the calling component. If a file with the same name exists in more than one directory, the adapter returns `true` and the full path of the first file found to the calling component. If the file name does not exist, or the directory does not exist, the adapter returns `false` to the calling component.
List	All the file names and directories that are specified in the request are returned to the calling component. If only the directory is specified, all the file names in the directory are retrieved and sent as a response to the calling component. If the specified directory does not exist, a RecordNotFound exception is sent to the calling component.

Table 1. Supported outbound operations (continued)

Operation	Result
ServerToServer FileTransfer	The specified file is transferred from one FTP server directory to another FTP server directory. After the file has been transferred successfully, `true` is returned to the calling component. Both the FTP servers have to support the ServerToServerFileTransfer operation and a connection must be established between the FTP servers and the workstation where the adapter is running. If the request does not contain all necessary information about the two servers, the adapter sends an FTPFileServerToServerFileTransfer exception to the calling component. **Note:** The ServerToServerFileTransfer operation does not support FTPS (FTP over SSL and FTP over TLS) or SFTP protocol.
ExecuteFTPScript	The commands contained in a FTP script file are run in the adapter workstation. The operation runs only the commands that are supported by the FTP server. If the operation fails, the adapter sends an FTPFileExecuteFTPScript exception to the calling component. The script file should not contain connection-related commands such as `open` because the adapter uses an established connection to run the commands. The directory should be specified in DirectoryPath and filename in FileName property. If the commands in the script file need to be run in a particular directory on the FTP server, then the script file must first contain the command to change to that directory. A list of commands runs and their reply strings are returned to the calling component. The adapter also supports parameter substitution in the FTP script file (replacing parameters %1, %2 with actual values). The values are sent as part of the request. **Note:** The script file should contain commands that are supported by the selected protocol.

Generating a file sequence during Create operations

Adapter for FTP supports the generation of a file sequence during an outbound Create operation. The FileSequenceLog property is introduced to specify the full path of the file where the sequences are stored.

A sequence file is a file used to store the sequence number. The adapter obtains the sequence number in this file for the current operation and increments the existing number by one and updates the file. When a sequence file is created, the file does not contain any data and the adapter starts generating the sequence number from 1.

For every request, the adapter reads the sequence number, increments it by 1 and then updates the sequence file. A sequence number is used while creating a request file in the target folder. If the number is not valid, for instance, if it is non-numeric, consists of special characters, or is zero or negative, the adapter starts the sequence again from 1. The adapter uses the existing sequence number in the file when the adapter is restarted.

Note: The sequence number is the only content in the sequence file that is used for an outbound create operation regardless of any directory or filename.

When a value is specified for the FileSequenceLog property, the adapter generates file sequence numbers, and appends to the file name of the files that it creates. The sequence number accepts the following format:

$FILENAME.$SEQUENCE_NUMBER.$FILE_EXT. For example, if HostName = localhost and Filename = Customer.txt, the output files are Customer.1.txt, Customer.2.txt, Customer.3.txt, and so on. The format is the same for all platforms, including z/OS® and i5/OS®. The sequence number continues to increment after multiple adapter restarts.

When the adapter is operating in a stand-alone mode, the value for the FileSequenceLog property should be in a file on the local file system. When the adapter is operating in a clustered environment, the value for the FileSequenceLog property should be in a file on the mapped drive that is accessible by all the clusters. The adapter must have write permission for the sequence log file or an IOException takes place.

Note: The file sequence number can be reset either by deleting the entry in the file or by deleting the file. A new sequence begins at 1. When the FileSequenceLog property and GenerateUniqueFilename property are both enabled, the GenerateUniqueFilename property value takes precedence, and the FileSequenceLog property is not generated.

You can generate the file sequence names. To generate file sequence names, specify:

1. The sequence file, which is the full path of the file where the sequence numbers are stored
2. The default target file name

The adapter generates a file name that consists of the default target file name with the sequence number appended to it. If the default file name has an extension, the sequence number is appended before the extension. For example, if the default file name is Customer.txt on the managed connection factory, the output file names that are created are Customer.1.txt, Customer.2.txt,, and so on.

The adapter performs the following steps to support compatibility with earlier versions:

1. The adapter reads the sequence file and checks for an entry of the form path = sequenceNumber.
2. If such an entry exists in the file, the sequence file contains the data in the form supported by Adapter for FTP version 6.1
3. The adapter gets the highest sequence number available from all the entries.
4. This number is used to create a new file.
5. The adapter increments the number and overwrites the entire file with the new number.

Note: Two different managed connection factories should not access the same sequence file. Also, two different adapter instances should not access the same sequence file unless they are part of a cluster, in which case they access a shared sequence file.

Generating unique file names

The Create operation supports the generation of unique file names under the following conditions:

- The Create operation supports the generation of unique file names when the GenerateUniqueFile property is set to true.

Note: For Append and Overwrite operations, the GenerateUniqueFile property is deprecated from v6.2 onwards. Even if the value is set for this property, the adapter considers the value as 'False'.

The properties that control the generation of unique file names are located in three places:

- The managed connection factory properties (the Default target file name and Sequence file properties)
- The interaction specification properties (the Default target file name and Generate unique file properties)
- The wrapper business object

The properties in the business object take precedence over the properties in the interaction specification, which take precedence over the managed connection factory properties. Unless you want to handle a particular object differently, use the properties on the managed connection factory to control the generation of file names.

If the FTP server supports the `STOU` command specified in RFC 1123, the adapter uses this server support to generate the unique file names.

If the FTP server does not support the `STOU` command, the adapter for FTP generates a unique file and creates it on the FTP servers. The format of the file created by the adapter is `F` followed by the combination of `TP` and random numbers. The number ranges between `0` and `99999`. The following examples illustrate this format: `FTP0, FTP9, FTP729, FTP99999`

Note: The adapter does not support both the GenerateUniqueFile and StagingDirectory options simultaneously.

Outbound data transformation

Data transformation during outbound communications refers to the process by which the adapter transforms business objects into an event record created in a native format, such as bytes or a string. The adapter uses adapter-specific data binding and data handlers to accomplish this.

Data transformation allows external applications to send and receive data in a format that they can understand and process easily. The data bindings and data handlers that the adapter uses to create the event record from the corresponding attributes in a business object are configured through the external service wizard in WebSphere Integration Developer.

Data bindings

Data bindings are essentially maps that define how a business object should be formatted. Data bindings are responsible for reading the fields in a business object and filling the corresponding fields in an event record. Each data binding is a map that defines how a business object should be formatted. The adapter for FTP uses the FTPFileBaseDataBinding data binding during outbound communication.

During outbound communications, the data binding uses the following fields in a business object, and populates their equivalent fields in an event record with their values:

- DirectoryPath
- Filename

- DataConnectionMode
- FileTransferType
- DataProtectionLevel
- SecondServerDirectory
- SecondServerUsername
- SecondServerPassword
- IncludeEndBODelimiter
- FileInLocalDirectory
- LocalDirectoryPath
- LocalArchivingEnabledForCreate
- LocalArchiveDirForCreate
- StagingDirectory
- GenerateUniqueFile
- SplittingFunctionClassName
- SplitCriteria
- DeleteOnRetrieve
- ArchiveDirectoryForRetrieve
- FileContentEncoding

For data that does not require transformation, the adapter conducts pass-through processing because data passes through the system without being altered.

Data handlers

In addition to data bindings, data transformation requires the use of a data handler. Data handlers perform the conversions between a business object and a native format. From version 6.2 onwards, the WebSphere Adapter for FTP provides the following data handlers:

- Delimited
- Fixed width
- XML

Inbound processing

The Adapter for FTP supports inbound processing of events. The adapter polls a file system associated with an FTP server for events at specified intervals. Each time a file is created in the event directory, the adapter tracks it as an event. When the adapter detects an event, it requests a copy of the file, converts the file data into a business object, and sends it to the consuming service.

The following illustration shows the inbound processing flow for WebSphere Adapter for FTP.

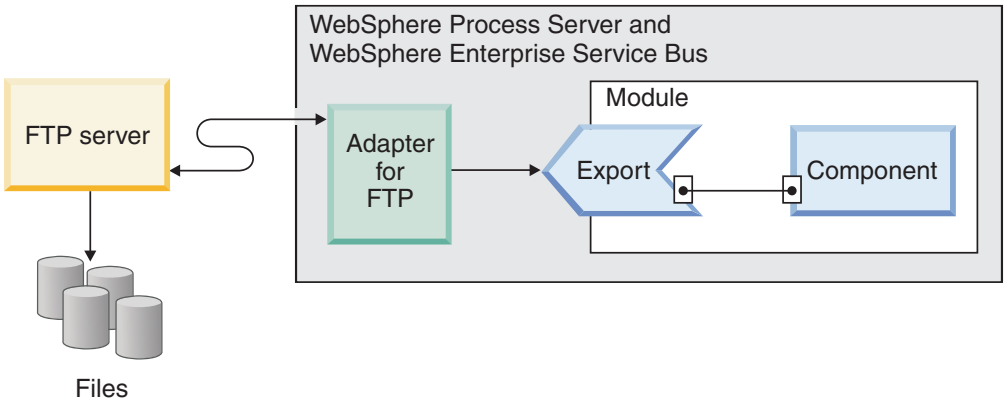


Figure 3. Inbound processing flow

The adapter polls files from the event directory of the FTP server at regular intervals based on the FTPPollFrequency property. When a file arrives in the event directory, the adapter reads the entire file and downloads the file to a local event directory on the adapter workstation. After the file is downloaded, the adapter either archives the file in the FTP server in an archive directory given by the FTPArchiveDirectory property or deletes it based on your configuration. The event directory, archive directory, poll frequency, and poll quantity (the number of files to poll in a single poll cycle) are all configurable properties.

Note: The value of event directory property should represent the absolute path of the directory.

After the business objects are successfully posted to the export, the events in the local staging directory are either archived in an archive directory on the local file system or deleted, based on your configuration. The adapter must archive or delete the events or they will be polled again.

Inbound event processing consists of the following steps:

1. FTP server generates events in the form of files.
2. The Adapter for FTP polls the event directory.
3. The files are downloaded to the adapter.
4. The files are split based on the SplittingFunctionClassName and SplitCriteria properties. The event file is split into several chunks and each chunk is posted to the export separately. This reduces memory loading during event processing.
   - If splitting is done based on a delimiter, the class that performs this function and the split criteria are provided.
   - If splitting is done based on file size, the class name that performs this function is provided.
   - If splitting is done based on other criteria, you must provide your own file splitting class.
5. The adapter sends the data, including the location of the polled document and the host name of the machine that the file was retrieved from, to the export through a function selector, where the configured data binding is invoked, to convert the text record into a business object.

Supported inbound operation

The adapter supports the emitFTPFile operation, which is taken as the default operation during inbound configuration.

Event file locking

File locking behavior is operating system dependent. In Windows®, if any of the files being polled by the adapter from the event directory are in use by another application and in the process of being copied to the event directory, they are not made available to the adapter for processing.

However, in UNIX® environments, such as AIX®, there is no file locking mechanism that prevents applications from accessing files that are being written to. A file that is being copied to the event directory by another application is made available to the adapter for processing, causing erroneous results. There is no platform-independent way in Java™ to check whether a file is being written to.

To prevent this situation from occurring, you can first copy the event file to a staging directory and then move it to the event directory using the move command. Some sample UNIX scripts are provided as part of the adapter. The script file named CheckIfFileIsOpen.sh is available in the Unix-script-file folder in the adapter installer.

Rule-Based filtering of events

The adapter supports the rule-based filtering of events, which is optional for inbound processing. You can filter the events based on multiple rules. You can define a combination of these rules, group them with Boolean logic, and filter the events using the following metadata:

- FileName
- File Size
- Last Modified

For example, you can use *FileName* "MatchesFilePattern" *.txt, where *FileName* is the property type, "MatchesFilePattern" is the operator and "*.txt" is the value.

Though using the rule is optional and specifying an event file mask is mandatory, the rule takes a higher precedence over the event file mask, when both a rule and an event file mask are specified. Event file mask is effective only when there is no rule specified. By default, an event file mask has "*.*" as the default value.

Rule-based filtering does not support the logical "OR" operator values between multiple rules.

Note: Adapter does not support rule-based filtering when the EIS is on MVS platform.

Table 2. Metadata filtering properties

Property	Valid operators	Value	Prerequisites
FileName	Matches_File_Pattern	For example: *.txt	Nil
	Matches_RegExp	Java Regular Expression	

Table 2. Metadata filtering properties (continued)

Property	Valid operators	Value	Prerequisites
FileSize	Greater than, Less than, Greater than or equal to, Less than or equal to, Equal to, Not equal to.	Numeric value in Bytes. For example: 10000	Nil
LastModified	Greater than, Less than, Greater than or equal to, Less than or equal to, Equal to, Not equal to. **Note:** Select 'Equal to' operator when you choose the days of week.	Day of the week or Time. For example : MONDAY or 20:41:10	Nil
END-OF-RULE	END-OF-RULE	END-OF-RULE	Nil

Function selectors

During inbound processing, a function selector returns the appropriate operation to be called on the service. You choose a function selector when you configure the adapter for inbound processing in the external service wizard. The adapter provides three function selectors, `FilenameFunctionSelector`, `EmbeddedNameFunctionSelector`, and `RootNameFunctionSelector`.

FilenameFunctionSelector

`FilenameFunctionSelector` is a rule-based function selector that provides object name resolution based on regular expressions that map to file names. A regular expression is a string that is used to describe or match a set of strings according to certain syntax rules.

The following table shows examples of matching rules, where a rule consists of the ObjectName and Rule fields.

Table 3. Examples of matching rules for FilenameFunctionSelector

FileName	ObjectName	Rule
`Customer0001.txt`	`Customer`	`CUST.*TXT`
`2231ORZ93.z21`	`Order`	`[0-9]*OR[A-Z][0-9]{2}.*`
`2231ORZ93.z21`	`Order`	`*OR.*`

Note that the rules in the second and third rows resolve to the same name, but the rule in the second row requires a specific sequence of numbers and letters in order for the file name to match, whereas the rule in the third row resolves anything with the characters “OR” in the file name. The character combination `".*"` indicates that any character may occur any number of times.

To generate the native function name, the function selector prepends `emit` to the object name that you provide. For example, if the object name is Customer, the function selector returns the function name emitCustomer. The object name should be the payload object name, for example, Customer or Order, and not the wrapper or business graph name. For pass-through scenarios, use FTPFile as the object name.

You can configure `FilenameFunctionSelector` with multiple rules, each containing an object name, and a regular expression to match against the file name. If more than one rule matches, the function selector returns the object name based on the

first matching rule. If no rule matches, the adapter generates an error. If no rules are present in the configuration, the function selector uses the function name emitFTPFile.

For a detailed explanation of the rules governing the use of regular expressions, see the Java Class Pattern documentation at https://java.sun.com/j2se/1.4.2/docs/api/java/util/regex/Pattern.html.

EmbeddedNameFunctionSelector

`EmbeddedNameFunctionSelector` is used for content-specific business objects, where the object name is embedded in the event file. It returns the function name based on the required content data, and not on the wrapper. For example, if the content-specific business object is CustomerWrapperBG, the function returned by the function selector is emitCustomer.

`EmbeddedNameFunctionSelector` has to be configured with a data handler. The data binding must be the adapter-specific WrapperDataBinding, and it should be configured to use the same data handler that is configured with the function selector.

RootNameFunctionSelector

`RootNameFunctionSelector` is used only for global elements in business objects, where the global element name is the root element name in the event XML file. It returns the function name based on the global element name. For example, if the global element name is CustomerType1, the function returned by the root name function selector is 'emit CustomerType1'.

`RootNameFunctionSelector` should be used only for global elements with XML Datahandler or UTF8XMLDatahandler.

Note: To use global Elements with Delimited Datahandler or FixedWidth Datahandler, you should use `FilenameFunctionSelector` instead of `RootNameFunctionSelector`.

`RootNameFunctionSelector` does not need any more configuration, as it does not depends on the data handler to get the correct function name.

Inbound data transformation

Data transformation during inbound communications refers to the process by which the adapter transforms an event record created in a native format, such as bytes or a string, into a business object. The adapter uses an adapter-specific data binding and data handlers to accomplish this.

The data bindings and data handlers that the adapter uses to read the contents of the event record and fill the corresponding attributes in a business object are configured using the external service wizard in WebSphere Integration Developer.

Data bindings

To take fields from an event record created in a native format, and populate a business object, the adapter needs a data binding. Data bindings are responsible for reading the event record fields and populating the corresponding fields in a business object. The adapter for FTP uses the FTPFileBaseDataBinding data binding during inbound communication.

During inbound communications, the data binding takes the following fields from an event record and populates the following business object attributes with their values:

- Filename
- ChunkInfo
- DirectoryPath
- FileContentEncoding
- FtpServerHostName
- FtpServerEventDirectory

For data that does not require transformation, the adapter conducts pass-through processing because data passes through the system without being altered.

Data handlers

In addition to data bindings, data transformation requires the use of a data handler. A data handler converts data from a native format into a business object. From version 6.2 onwards, the WebSphere Adapter for FTP provides the following data handlers:

- Delimited
- Fixed width
- XML

Passing files by reference

The adapter also supports a PassByReference feature, where only the event file name is sent to the export. The event file is appended with a time stamp and is available in the local archive directory. This feature is used when data transformation is not necessary.

Splitting files

The inbound event processing mode supports an optional file splitting feature, where the event file is split into several business objects, also known as chunks, and each business object is posted to the export separately. This reduces memory loading during event processing. File splitting is performed based on either a delimiter or on a file size specified in the SplitCriteria property.

The adapter provides SplitBySize and SplitByDelimiter classes for file splitting. Optionally, you can provide a custom file splitter class and use it by inputting the class name into the SplittingFunctionClassName property.

Splitting files by size

The size value is set in the SplittingFunctionClassName property.

Chunks refer to the resulting files after file splitting is performed. When chunking is enabled, each chunk of the file is posted to the export separately. The number of business objects that are specified in the PollQuantity property is posted to the export. For example, If the value for PollQuantity is 3, then:

The number of business objects polled is 3.

The number of business objects received by the export is 3.

The adapter does not reassemble chunked data. It provides the information about the chunked data for an external application to merge the chunks. The chunking information is set in the chunkInfo property, which is contained in the business object. This information includes the chunk size in bytes, and the event ID. An example of an event ID is:

```
AbsolutePathOfTheEventFileNameInLocalEventDirectory_/_yyyy_MM_dd_HH_mm_ss_SSS.
currentBONumber_/_totalBOs
```

Splitting files by delimiter

Delimiters are specified values, used for splitting event files. The delimiter is specified in the SplitCriteria property.

The following rules apply to the use of delimiters:

- The specified delimiter must not be the same as any of the data contained within the business object. If it is the same, file splitting can produce incorrect results.
- The delimiter must contain the exact value of new line representation in the event file. If the event file is created on a MAC machine, the new line character is `\r`. On UNIX machines it is `\n`, and for windows machines it is `\r\n`.
- If there is more than one delimiter, each delimiter must be separated by a semicolon `(;)`. If the semicolon is part of the delimiter, the semicolon must be represented as `\;`. For example, if the delimiter is `##\;##` then it is processed as `##;##`, which means that the semicolon is part of the delimiter.
- To skip content that is part of the delimiter, specify a double semicolon `(;;)` in front of it so that the content between the delimiters is skipped. For example, if the event file contains a business object in the following format and the delimiter is `##;;$$`, then:

  ```
  Name=Smith
  Company=IBM
  ##this is the content that will be skipped by the adapter$$
  ```

  The adapter will consider `##$$` as the delimiter and skips "`this is the content that will be skipped by the adapter`."
- The delimiter takes any value and there are no restrictions. The following are valid delimiter examples:
  - `####;\n;\n`
  - `####;$$$$;\n;####`
  - `%%%%;$$$$$;#####`
  - `\n;\n;$$$$`
  - `####\;####;\n;$$$$$`
  - `\n;\n;\n`
  - `####;;$$$$`
  - `\r`
  - `\r\n`
  - `$$$$;\r\n`
- If the delimiter is located at the end of the file, the SplitCriteria property uses `END_OF_FILE` to determine the physical end of the file.
- When each business object record in an event file is separated by a valid delimiter and if there is no delimiter or an invalid delimiter for the last business object record, the adapter will still be able to process the business object records.

- During inbound processing and splitting of the event file based on a delimiter, if the business object records present in an event file are separated by a delimiter and this delimiter is present at the beginning of each record instead of end of the record then the adapter considers that the delimiter is always present at the beginning of each record and processes accordingly.

Example 1:

```
John Doe,123,Washington Ave,222-123-4567
Jane Smith,234,Washington Ave,222-123-4568
```

The separator is the end of line character. In this example you would specify `\r\n` for Windows, `\r` for MAC, and `\n` for UNIX.

Example 2:

```
John Doe
123 Washington Ave
222-123-4567
####
Jane Smith
234 Washington Ave
222-123-4568
```

The separator is `####`.

Example 3:

```
ISAJohnDoe1*IBM***********USA***********
ISAJohnDoe2*IBM***********USA***********
ISAJohnDoe3*IBM***********USA***********
```

Content in an event file with a delimiter (ISA) at the beginning of each record.

Event recovery

The adapter supports event recovery for inbound processing in case of abrupt termination. During event processing, the adapter persists the event state in an event persistence table located on the data source. You must set up this data source before you can create the event persistence table.

To use the recovery feature provided by WebSphere Process Server, you must set the AssuredOnceDelivery property in the activation specification to `true`. If it is set to `false`, failed events cannot be recovered. Duplicate events can be delivered if AssuredOnceDelivery is set to `false`. To improve performance, you can set the AssuredOnceDelivery properties to `false`.

Event persistence table

The event persistence table is a persistent cache where events are saved until the adapter can process them. The adapter uses event persistence tables to keep track of inbound requests as they make their way through the system. Each time a file is created, updated, or deleted, the adapter tracks the activity as an event and updates the status of the event in the event persistence table. The status of each event is continually updated by the adapter for recovery purposes until the events are delivered to a configured export.

If the adapter detects that there is no event persistence table, it automatically creates one when the module is deployed to the runtime environment. Each event

persistence table created by the adapter is associated with a specific inbound module. The adapter does not support multiple adapter modules pointing to the same event persistence table.

When the adapter polls the FTP server, it creates an entry in the event persistence table for each event that matches the search criteria specified in the activation specification properties. The adapter records the status of each new entry as `NEW`. When the adapter copies the event from the FTP server to the in-progress folder on the local system, it marks the entry as `IN PROGRESS`. When the adapter sends the event to the function selector for data transformation, it deletes the entry from the event table.

Note: When guaranteed event delivery is not required, the adapter can poll for events without the existence of an event persistence table.

The following table describes each event persistence table value.

Table 4. Event persistence table structure

Column Name	Type	Description
EVNTID	Varchar(255)	A unique event ID for tracking purposes. The adapter uses this ID to track events during inbound processing.
EVNTSTAT	integer	The status of the event. The adapter uses the status to determine whether an event is new or in process. Event status values: **NEWEVENT (0)** The event is ready to be processed. **PROCESSED (1)** The adapter successfully processed and delivered the event. **FAILED (-1)** The adapter was unable to process this event due to one or more problems.
XID	Varchar(255)	Used by the adapter for assured event delivery and recovery.
EVNTDATA	Varchar(255)	Used by the adapter to mark the failed events as `ARCHIVED` to ensure that they are not processed again during adapter startup or recovery.

Event archive

Archived events are stored in the archive directory with a file extension that is specified in the FTPRenameExt property. Event archiving is an optional feature, which provides you with a record of all the events that have been processed. You can use this information to review whether the events were processed successfully.

Event archiving is used differently in different configurations:

- When both the FTPArchiveDirectory and FTPRenameExt values are provided and FTPRenameExt is set to `processed`, the archived file is located in the specified archive directory with the following syntax:
  `filename_timestamp``.processed`

- When only the FTPArchiveDirectory value is provided, the archived file is located in the specified archive directory in the following syntax: *filename_timestamp*
- When neither the FTPArchiveDirectory nor the FTPRenameExt values are provided, the event file is deleted from the event directory of the FTP server after the file is successfully downloaded to the local event directory.
- When only the FTPRenameExt value is provided and is set to `processed`, the archived file is located in the event directory of the FTP server with the following syntax: *filename_timestamp*`.processed`

Archiving on MVS platforms

Multiple Virtual Storage (MVS) operating systems do not support special characters such as an underscore in data set or recordset names. On Windows and UNIX platforms, use a time stamp in the original file name while archiving the file. This prevents duplicate file names in an archive folder, thereby preventing the overwriting of an existing file. Use the following format for MVS systems:

```
Event File: Test Archived

file: Test.TSyyyyMM.TSDDHHMM.TSSsSss
```

Where:

```
yyyy -- year

MM -- month

DD -- date

HH -- hour

MM -- minutes

Ss -- seconds

Sss -- milliseconds
```

The data set or recordset separator is **.** (decimal) on MVS platforms. The maximum number of **.** (decimals) allowed in a data set or recordset is six. The data set or recordset name must not exceed eight characters per **.** (decimal), and the total number of characters must not exceed 44. An example of a file name in this format is:

```
FTPRenameExt: ARCHIVE

Archived File: TEST.TS200304.TS290535.TS42234.ARCHIVE
```

Business objects

A business object is a structure that consists of data, the action to be performed on the data, and additional instructions, if any, for processing the data. The data can represent either a business entity, such as an invoice or an employee record, or unstructured text.

How the adapter uses business objects

The adapter uses business objects to send data to or obtain data from the FTP server. The adapter's main job during inbound operations is to take information from an event record created in a native format, convert it to a business object, and forward it to a service. For outbound operations, this process happens in reverse. The adapter receives a business object from a service, creates an event record from the details it finds in the business object, and then sends the event record to the FTP server.

How data is represented in business objects

Business objects are created using the business object editor in WebSphere Integration Developer, which provides a graphical view of your business objects. As shown in the following illustration, a business object consists of a set of fields and their values. This is a customer business object. As you can see, it records name, address, and phone number information for a customer record. This example uses string values, but many other values are supported by the business object editor.

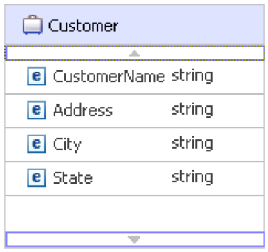


Figure 4. How data is represented in business objects

How business objects are created

You can create business objects by using either the external service wizard or the business object editor, both of which can be launched from WebSphere Integration Developer.

If you have defined XSD files using the business object editor prior to starting the external service wizard, the adapter will create business objects from these schemas. For instructions on how to use the business object editor to create business objects, refer to the following link: http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp. After you create your business objects, you can use the business object editor to define the hierarchy of the business objects.

Business graphs

You can optionally choose, during adapter configuration, to generate a business graph. In version 6.0.2, each top-level business object is contained in a business graph, which includes a verb that an application can use in version 6.0.2 to specify additional information about the operation to be performed. In Version 6.2.x, business graphs are optional; they are required only when you are adding business objects to a module created with a version of WebSphere Integration Developer earlier than Version 6.2.x. If business graphs exist, they are processed, but the verb is ignored.

Global elements

Global Elements are the globally defined schema elements, which can be reused by referencing them in other parts of the schema or from other schema documents.

The Adapter for FTP supports global elements in structured business objects. The adapter supports global elements of anonymous type and global elements of named type, with namespace as well as without namespace in schema business objects.

For more information see, "Global elements in a structured business object" on page 146.

WebSphere Application Server environment variables

When you configure the adapter for inbound or outbound processing using the external service wizard, you set values for various required local files and directories. You can later change these values in the deployed application from the WebSphere Process Server administrative console.

With WebSphere Process Server version 6.2 and onwards, instead of hard coding values for directories and files, you can declare them as WebSphere Application Server environment variables, and specify the environment variable names when you run the external service wizard. When you deploy your application, the environment variable name is replaced with the actual value and used by the adapter. If you want to change the property value, you can change the environment variable in the WebSphere Process Server administrative console.

WebSphere Application Server environment variables can be used for all string property values (not Boolean or integer variables) that are set in inbound and outbound configuration.

You define WebSphere Application Server environment variables using the WebSphere Process Server administrative console.

When you define a WebSphere Application Server environment variable, you specify:

- The name of the environment variable, for example, EVENT_DIRECTORY.
- The value that the symbolic name represents, for example: C:\ftp\event
- The scope for the environment variable. The scope level determines the level at which the environment variable is visible in the administrative console. The scope level can be server, node, or cell:
  - Server scope limits visibility to the named server. The server scope is the most specific scope for defining environment variables.
  - Node scope limits visibility to all the servers on the named node. This is the default scope.
  - Cell scope limits visibility to all servers on the named cell.

See the topic Defining WebSphere Application Server environment variables "Defining WebSphere Application Server environment variables" on page 71 for detailed information about how to create a WebSphere Application Server environment variable.

External service wizard

The external service wizard in WebSphere Adapter for FTP is used to create services and to generate business objects from the selected objects. The wizard also generates the service artifacts that enable the adapter to run as a Service Component Architecture (SCA) component.

Support for the Log and Trace Analyzer

The adapter creates log and trace files that can be viewed with the Log and Trace Analyzer.

The Log and Trace Analyzer can filter log and trace files to isolate the messages and trace information for the adapter. It can also highlight the adapter's messages and trace information in the log viewer.

The adapter's component ID for filtering and highlighting is a string composed of the characters `FTPRA` plus the value of the adapter ID property. For example, if the adapter ID property is set to `001`, the component ID is `FTPRA001`.

If you run multiple instances of the same adapter, ensure that the first eight characters of the adapter ID property are unique for each instance so that you can correlate the log and trace information to a particular adapter instance. By making the first seven characters of an adapter ID property unique, the component ID for multiple instances of that adapter is also unique, allowing you to correlate the log and trace information to a particular instance of an adapter. For example, when you set the adapter ID property of two instances of WebSphere Adapter for FTP to `001` and `002`. The component IDs for those instances, `FTPRA001` and `FTPRA002`, are short enough to remain unique, enabling you to distinguish them as separate adapter instances. However, instances with longer adapter ID properties cannot be distinguished from each other. If you set the adapter ID properties of two instances to `Instance01` and `Instance02`, you will not be able to examine the log and trace information for each adapter instance because the component ID for both instances is truncated to `FTPRAInstance`.

For outbound processing, the adapter ID property is located in both the resource adapter and managed connection factory property groups. If you update the adapter ID property after using the external service wizard to configure the adapter for outbound processing, be sure to set the resource adapter and managed connection factory properties consistently. It prevents inconsistent marking of the log and trace entries. For inbound processing, the adapter ID property is located only in the resource adapter properties, so this consideration does not apply.

For more information about the adapter ID property, see “Adapter ID (AdapterID)” on page 154. For more information about the Log and Trace Analyzer, see http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/topic/org.eclipse.hyades.log.ui.doc.user/concepts/cltaviews.htm.

Business faults

The adapter supports business faults, which are exceptions that are anticipated and declared in the outbound service description, or import. Business faults occur at predictable points in a business process, and are caused by a business rule violation or a constraint violation.

Although WebSphere Process Server and WebSphere Enterprise Service Bus support other types of faults, the adapter generates only business faults, which are

called *faults* in this documentation. Not all exceptions become faults. Faults are used only when the outbound operations are configured with response type. Faults are generated for errors that are actionable, that is, errors that can have a recovery action that does not require the termination of the application. For example, the adapter generates a fault when it receives a business object for outbound processing that does not contain the required data or when the adapter encounters certain errors during outbound processing.

Note: The faults for a particular operation will be enabled only if that operation has a response configured.

Fault business objects

The external service wizard creates a business object for each fault that the adapter can generate. In addition, the wizard creates a WBIFault superset business object, which has information common to all faults, such as the message, errorCode, and primaryKeySet attributes as shown in Figure 5.

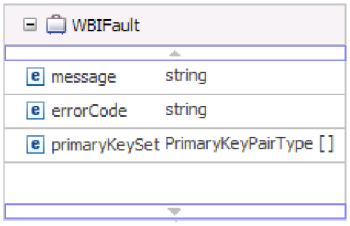


Figure 5. The structure of the WBIFault business object

The WebSphere Adapter for FTP enables faults for you. Manual configuration of faults is not required.

Chapter 2. Planning for adapter implementation

To implement the IBM WebSphere Adapter for FTP, you must plan for inbound and outbound processing and consider security and performance requirements.

Before you begin

Before you begin to set up and use the adapter, you should possess a thorough understanding of business integration concepts, the capabilities, and requirements of the integration development tools and runtime environment you use.

To configure and use WebSphere Adapter for FTP, you should understand and have experience with the following concepts, tools, and tasks:

- The business requirements of the solution you are building.
- Business integration concepts and models, including the Service Component Architecture (SCA) programming model.
- The capabilities provided by the integration development tools you use to build the solution. You should know how to use the tools to create modules, test components, and complete other integration tasks.
- The capabilities and requirements of the runtime environment you use for the integration solution. You should know how to configure and administer the host server and how to use the administrative console to set and modify property definitions, configure connections, and manage events.
- The File Transfer Protocol (FTP), the protocol for exchanging files over the Internet.
- The FTP server being used to access the files on a specific file system in your solution.

Security

For secure communication, Secure Socket Layers (SSL) can be configured to protect the integrity of information being passed between the FTP server and the adapter. For users who require it, the adapter can be configured to run in support of the Federal Information Processing Standard (FIPS) 140-2. The adapter also supports FTP over SSH (SFTP), which is a network protocol that runs on a secure SSH channel on port 22.

Support for secure FTP

Data that travels across a network can be intercepted by third parties. When this data includes private information, such as passwords or credit card numbers, steps should be taken to make this data unintelligible to unauthorized users. Data encryption can be achieved using cryptographic protocols, such as secure socket layer (SSL) and transport layer security (TLS). When FTP protocol is used with SSL or TLS, the security mechanism is referred to as secure FTP or FTPS (Also known as FTP over SSL or FTP over TLS).

By configuring secure socket layers (SSL) or transport layer security (TLS), you protect the integrity of information sent between the FTP server and adapter. When the adapter is configured to work in secure FTP, both the control connection and data connection can be encrypted.

Secure socket layer (SSL)

Secure socket layer (SSL) is a network protocol used to transmit data in a secure mode. SSL protocol uses the public key cryptography technique to encrypt the data while transferring, and also ensures data confidentiality.

Transport layer security (TLS)

Transport layer security (TLS) is a protocol used for secure data transfer between the client and the server. It is the successor of the secure socket layer (SSL) protocol.

FTPS connection modes

The FTPS client can establish a connection with the secure FTP server in either implicit or explicit mode.

Implicit mode: In an implicit mode, the communication between the client and server is set up immediately in secure mode. The text information exchanged between the client and server is in an encrypted format. The default port for implicit mode is 990.

Explicit mode: In an explicit mode, the connection begins with an unencrypted FTP connection. When any sensitive information, such as password, needs to be sent, the client explicitly issues a request to switch to a secure FTP connection. After the successful SSL negotiation, a secure command channel is established between the client and the server.

Explicit mode works with the default port 21 and is compliant with RFC 2228 commands. RFC 2228 specifies the mechanism for authenticating connections and confidential data transfer between the client and server, and this is referred to as explicit mode. The AUTH command is used for specifying the security mechanism for the explicit mode. The client sends an AUTH command (AUTH SSL/TLS) to the FTPS server and switches to a secure command connection.

By using the connection modes, the data protection level with which the data is transferred between the client and the server can be configured.

Data connection encryption

According to RFC 2228, Protection buffer size (PBSZ) and data channel protection level (PROT) commands are issued by the client to specify the protection level on the data channel.

Protection buffer size (PBSZ) is used to negotiate a maximum protected buffer size for the data connection. PBSZ command accepts a long value as an argument, and determines the maximum size of the buffer in which the encoded data is sent or received during data transfers.

FTP over TLS supports only PBSZ 0 to ensure that the buffering of data does not takes place. PBSZ command with the argument value '0' indicates a streaming protocol and the data is transferred as a stream of data.

PROT command allows client or server negotiation for the security level data connection. RFC 2228 specifies the following four levels of protection:

1. Clear (C): The Clear protection level indicates that the data channel will carry the raw data for the file transfer, with no security applied.
2. Safe (S): The Safe protection level indicates that the data is integrity protected.
3. Confidential (E): The Confidential protection level indicates that the data is confidentiality protected.
4. Private (P): The Private protection level indicates that the data is integrity and confidentiality protected.

FTP over TLS protocol supports only Clear and Private levels of data protection.

Server authentication

Server authentication is a check performed for a secure connection. While establishing a SSL connection to the FTPS server, the FTP client performs a server certificate validation against the certificates present in the client trust store. The client trust store contains the certificates of all servers that are trusted. If the required certificate of the server is found in the client trust store, then a connection is established.

If the certificate is not found in the client trust store, the server is considered as an untrusted server, an exception is generated, and a connection is not established with the FTPS server.

Client authentication

Client authentication is similar to server authentication, except that the server requests a certificate from the client to verify if it is from a trusted client. The certificate has to be signed by a certificate authority trusted by the server. The client authentication requires a compatible FTPS server for authenticating. When a server requests a certificate, the client has the option to send a certificate. The server allows the connection if the client's certificate can be trusted.

The FTP server authenticates the client based on the public certificate while establishing a SSL connection. The client provides the public key during a SSL connection and is exchanged with the FTPS server, which authenticates the client's identity based on the certificates configured in the server's trusted certificates.

Configuring the adapter for secure FTP (secure socket layers or transport layer security)

WebSphere Adapter for FTP supports connecting to a secure FTP server (FTPS) using either the SSL or TLS protocol. WebSphere Adapter for FTP can be configured to connect to the FTPS server in either explicit or implicit mode. The adapter supports secure FTP using SSL v3.0 and TLS v1.0

Before you begin

To enable SSL, ensure that the following prerequisites are met:

- The FTP server supports secure communication using SSL.
- The FTP server has its own private key and certificate.
- The adapter uses a passive FTP mode of data transfer with a secure FTP server. If there is a firewall between the client and the server, the firewall settings might need to be configured to enable this mode.

The data connection protection commands are exchanged between the adapter and the server after you have successfully logged in but before you establish the data connection.

Note:

1. By default, the adapter issues PBSZ 0 command before issuing the PROT command.
2. The WebSphere Adapter for FTP supports Clear and Private levels of data channel protection.

Refer to the following configuration table that represents the different combinations.

Table 5. Configuration information

Configuration	Protocol	FTPS connection mode	Data connection encryption	Description
1	FTP over SSL	Implicit	Clear	With this configuration, the adapter connects to the FTP server in SSL implicit mode and the data is transferred in the clear text format and there is no data encryption.
2	FTP over SSL	Implicit	Private	With this configuration, the adapter connects to the FTP server in SSL implicit mode and the data channel is encrypted.
3	FTP over SSL	Explicit	Clear	With this configuration, the adapter connects to the FTP server in SSL explicit mode and the data is transferred in the clear text format. There is no data encryption.
4	FTP over SSL	Explicit	Private	With this configuration, the adapter connects to the FTP server in SSL explicit mode and the data channel will be encrypted.
5	FTP over TLS	Implicit	Clear	With this configuration, the adapter connects to the FTP server in TLS implicit mode and the data is transferred in clear text format. There is no data encryption.
6	FTP over TLS	Implicit	Private	With this configuration, the adapter connects to the FTP server in TLS implicit mode and the data channel is encrypted.
7	FTP over TLS	Explicit	Clear	With this configuration, the adapter connects to the FTP server in TLS explicit mode and the data channel is in clear text format. There is no data encryption.

Table 5. Configuration information (continued)

Configuration	Protocol	FTPS connection mode	Data connection encryption	Description
8	FTP over TLS	Explicit	Private	With this configuration, the adapter connects to the FTP server in TLS explicit mode and the data channel is encrypted.

About this task

Files passing through the FTP server are vulnerable to third-party interference when SSL is not configured for use with the adapter. Using SSL prohibits data from being modified intentionally or unintentionally during transport and protects it from being intercepted. SSL is effective because it uses several cryptographic processes: public key cryptography for authentication with the FTP server and secret key cryptography and digital signatures for privacy and data integrity. SSL allows the adapter to authenticate the identity of the FTP server.

Procedure

1. In the external service wizard, set the Protocol to `FTP` over SSL - File Transfer Protocol over Secure Socket Layer or `FTP` over TLS - File Transfer Protocol over Transport Layer Security.
2. In the Secure configuration area of the external service wizard, set the FTPS connection mode to either `Explicit` or `Implicit` mode. The default port number used for Explicit mode is 21 and Implicit mode is 990. Change the port number accordingly if the FTPS server runs on a different port.
3. Set the **Data channel protection level** to `Private` or `Clear`. If you select the:
   - Private level of data protection, the data transfer will be integrity and confidentiality protected
   - Clear level of data protection, the data transfer will be in clear form.

   Note: The default value is set to private.
4. Set the FTP adapter trust store. A trust store helps a FTP client decide what it can trust. While using SSL, FTPS server sends its certificate to the FTP client for verification. The FTP client verifies the certificate to ascertain that it is communicating with the intended FTP server. To enable this verification process, the FTP server's certificate should be present in the client's trust store.
   a. Use keytool utility, if you want to import server's certificate into client's trust store. For example, enter the command keytool -import -v -alias serverCert -file server.cert -keystore clientTrustStore where server.cert is the certificate of the server and clientTrustStore is the trust store of the client.
   b. Set the **Keystore type** to the type of keystore used while creating the truststore.
   c. Set the **Truststore file** to the absolute path of the truststore file.
   d. Set the **Truststore password** to the password of the truststore. The password is used to check the integrity of the contents of the truststore.
5. Optional: Client authentication can be enabled while establishing a SSL connection. When using SSL, FTPS server requests for the client's certificate. The FTPS server verifies the certificate sent by the client to ascertain that it is communicating with the intended client. To enable this verification process, the FTPS server has to support client authentication and the client's certificate

should be present at the server's trust store. At the client's end, client's keystore information has to be available for the exchange of the certificate to take place.

a. You can create a keystore using the keytool utility.
b. Set the Keystore file to the absolute path of the keystore.
c. Set the Keystore password to the password of the keystore. The password is used to check the integrity of the contents of the keystore
d. Set the Key password to the password provided while creating the key in the keystore. This value is required to extract the certificate from the keystore while establishing a SSL connection.

   Note: Ensure that the value of Keystore type property is same as the type used while creating the keystore.

Configuring the adapter for federal information processing standard 140-2

The federal information processing standard 140-2 (FIPS) is a United States government standard for cryptographic features like encryption, decryption, hashing (message digests), secure socket layers, transport layer security, Internet Protocol security, Secure shell, signatures, key exchange, and key or certificate generation used in software products and modules. If you are an user working with the United States government who must conform to the FIPS standard, you can configure the adapter to run in FIPS mode.

About this task

Configuring the adapter to run in FIPS mode restricts the adapter working with modules whose cryptographic features comply with FIPS approved methods and providers. From an adapter perspective, running in FIPS mode restricts the adapter using the transport layer security (TLS) secure socket protocol. A single Java Virtual Machine (JVM) cannot be in FIPS mode and contain non-FIPS mode JSSE applications, executing at the same time.

Note: For the adapter to run in FIPS mode, the FTP server must support SSL v3.1, which is the same as TLS v1.0, and it must be enabled through the wizard of the FTP server. If not properly supported by SSL v3.1, the SSL handshake with the adapter may fail.

When in FIPS 140-2 mode, IBM WebSphere Adapter for FTP uses the FIPS 140-2 approved cryptographic provider(s); IBMJCEFIPS (certificate 376) and IBMJSSEFIPS (certificate 409). The certificates are listed on the NIST Web site at http://csrc.nist.gov/cryptval/140-1/1401val2004.htm

To run the adapter in FIPS mode, you must instruct the adapter to use the IBM Java Secure Socket Extension (IBMJSSE2) provider package. The IBMJSSE2 provider is the preregistered Java secure socket extension provider in the Java security file in IBM SDK, version 6.0. IBMJSSE2 uses FIPS-approved packages.

Note: The Secure Socket Layer (SSL) is not supported in FIPS mode.

Complete the following steps to run the adapter in FIPS mode:

Procedure

1. In the IBMJSSE2 provider, set the com.ibm.jsse2.JSSEFIPS property to `True`.
   a. Follow the steps to configure the values:

- Invoke WebSphere Process Server admin console by connecting to http://<hostname>:<portnumber/ibm/console/. For example, http://9.186.116.151:9060/ibm/console/
- Navigate to Servers.
- Select WebSphere application servers from Server Types.
- Select Configuration, Server Infrastructure, Java and Process Management, and Process Definition.
- Select Additional properties, Java Virtual Machine, and Custom properties.
- Click **New** and set the **Name** to com.ibm.jsse2.JSSEFIPS.
- Set the **Value** to true.

2. Set the following security properties so that the IBMJSSE2 provider will handle all JSSE requests.
   a. Set the ssl.SocketFactory.provider property to com.ibm.jsse2.SSLSocketFactoryImpl.
   b. Set the ssl.ServerSocketFactory.provider property to com.ibm.jsse2.SSLServerSocketFactoryImpl.
   c. Follow the steps to configure the values:
   - Invoke <jave-home>/lib/security/java.security, where <java-home> is the home path of the WebSphere Process Server's Java Virtual Machine (JVM). For example, C:\IBM\WebSphere\ProcServer\java\jre\lib\security\java.security
   - Open the file, java.security, and find the segment similar to the listed one.

     ```
     # Default JSSE socket factories
     #ssl.SocketFactory.provider=com.ibm.jsse2.SSLSocketFactoryImpl
     #ssl.ServerSocketFactory.provider=com.ibm.jsse2.SSLServerSocketFactoryImpl
     # WebSphere socket factories (in cryptosf.jar)
     ssl.SocketFactory.provider=com.ibm.websphere.ssl.protocol.SSLSocketFactory
     ssl.ServerSocketFactory.provider=com.ibm.websphere.ssl.protocol.SSLServerSocketFactory
     ```

   - Uncomment the default JSSE socket factories and comment the WebSphere socket factories. The settings will be displayed as follows:

     ```
     # Default JSSE socket factories
     ssl.SocketFactory.provider=com.ibm.jsse2.SSLSocketFactoryImpl
     ssl.ServerSocketFactory.provider=com.ibm.jsse2.SSLServerSocketFactoryImpl
     # WebSphere socket factories (in cryptosf.jar)
     #ssl.SocketFactory.provider=com.ibm.websphere.ssl.protocol.SSLSocketFactory
     #ssl.ServerSocketFactory.provider=com.ibm.websphere.ssl.protocol.SSLServerSocketFactory
     ```

3. In the security properties file, add the IBMJCEFIPS provider com.ibm.crypto.fips.provider.IBMJCEFIPS to the provider list above the IBMJCE provider. Follow the *security.provider.n=providername* format where *n* denotes the order of the provider. The provider with a value of 1 is considered before the provider with a value of 2. Do not remove the IBMJCE provider.
   a. Follow the steps to configure the values:
   - Invoke <jave-home>/lib/security/java.security, where <java-home> is the home path of the WebSphere Process Server's JVM. For example, C:\IBM\WebSphere\ProcServer\java\jre\lib\security\java.security
   - Open the file, java.security, and find the segment similar to the listed one. The list displays the providers and their preference orders.

     ```
     #security.provider.1=com.ibm.crypto.fips.provider.IBMJCEFIPS
     security.provider.1=com.ibm.crypto.provider.IBMJCE
     security.provider.2=com.ibm.jsse.IBMJSSEProvider
     security.provider.3=com.ibm.jsse2.IBMJSSEProvider2
     security.provider.4=com.ibm.security.jgss.IBMJGSSProvider
     security.provider.5=com.ibm.security.cert.IBMCertPath
     ```

4. Edit the java.security file to insert the IBMJCEFIPS provider (com.ibm.crypto.fips.provider.IBMJCEFIPS) before the IBMJCE provider, and also renumber the other providers in the provider list.
   - If the provider exists, uncomment the line, `com.ibm.crypto.fips.provider.IBMJCEFIPS` and ensure that it is set before the line, `com.ibm.crypto.provider.IBMJCE`
   - After you made the settings, the file is displayed as follows:

```
security.provider.1=com.ibm.crypto.fips.provider.IBMJCEFIPS
security.provider.2=com.ibm.crypto.provider.IBMJCE
security.provider.3=com.ibm.jsse.IBMJSSEProvider
security.provider.4=com.ibm.jsse2.IBMJSSEProvider2
security.provider.5=com.ibm.security.jgss.IBMJGSSProvider
security.provider.6=com.ibm.security.cert.IBMCertPath
```

What to do next

For more details on configuring security details, see the security documentation for WebSphere Process Server or WebSphere Enterprise Service Bus.

Support for SFTP

SFTP, is a protocol that uses Secure shell (SSH) to transfer files. Unlike standard FTP, it encrypts both commands and data, preventing passwords and sensitive information from being transmitted over the network. It is functionally similar to FTP, but because it uses a different protocol, you cannot use a standard FTP client to talk to a SFTP server, or connect to a FTP server with a client that supports only SFTP.

Server verification

Server verification is a method where the client verifies the identity of the server before establishing the connection.

The adapter performs the server verification when the SFTP protocol is enabled. The adapter checks the SFTP server that it is trying to establish a connection with to see whether it is a trusted server or not.

Server verification requires a host key file as the input. The host key file must be available on the adapter workstation with the host keys of the trusted servers added to it. The entries in the host key file have to adhere to OpenSSH format of the KNOWN_HOSTS file.

The adapter verifies the server by comparing the host key presented by the server with the host keys present in the host key file. The adapter connects to the server only if the host key of the server is available in the host key file. If the host key of a trusted server is different from the one that is present in the host key file, then the host key entry in the host key file has to be modified to reflect the new entry.

If the server is not trusted (the host key is not present in the host key file), the adapter does not connect to the server, and the connection request fails, indicating that a connection was attempted to a non-trusted server and connection cannot be established due to security reasons.

Note: If you want to provide both the host name and the IP address, specify them, separating them with a comma, in the host key entry.

Public key authentication

Public key authentication is one of the most secure methods used to authenticate when using a Secure Shell. Public key authentication uses a pair of computer generated keys, one public and one private. The public key can be distributed and resides in the SFTP server. The private key is unique to the user and must not be shared.

The following properties are required to enable public key authentication:

- Host name
- Port number
- User name
- Private key

Passphrase is an optional property that is used to provide extra protection for the private key.

The key-pair can be generated using any third-party service and you can choose any of the standard encryption algorithms. The most commonly used algorithm is RSA; however, other algorithms such as DSA can be used.

Note: The key-pair must be in the OpenSSH format.

If both the Password (user name and password authentication) and the Private key (public key authentication) values are specified in the external service wizard, the Private key property value takes precedence. The adapter then tries to authenticate to the server using public key authentication.

Configuring the adapter for SFTP

SSH over FTP (SFTP) is a network protocol that provides a mechanism for file transfer over a reliable data stream. SFTP runs on a secure SSH channel on port 22 and encrypts all traffic using either user name and password authentication or public key authentication. Public key authentication uses a pair of computer generated keys, one public and one private.

About this task

Configure the adapter for FTP to work with a secure SSH server:

Procedure

1. Install and configure your SSH server. There are various SSH servers to choose from. Install and configure your selected server using the provider-specific installation information.
2. See either Outbound or Inbound **Setting deployment and runtime properties** to select Protocol as **SFTP - Secure shell (SSH) File Transfer Protocol** and specify the SFTP server connection and security information in the external service wizard.

Results

You have configured the adapter for SFTP.

Support for confidential logging and tracing

The adapter provides the ability to protect sensitive or confidential data in log and trace files from being seen by others without authorization.

Log and trace files for the adapter can contain data from your FTP server, which might contain sensitive or confidential information. Sometimes it is necessary for these files to be seen by individuals without authorization to view to sensitive customer data, for example, a support specialist who must use the log and trace files to troubleshoot a problem.

To protect the data in such situations, the adapter provides the HideConfidentialTrace property. The HideConfidentialTrace property specifies whether you want to prevent confidential user data from displaying in the adapter log and trace files. When this property is enabled, the adapter replaces the confidential data with XXX.

The following types of information are considered potentially sensitive data and are hidden:

- The contents of a business object
- The contents of an event record
- User ID
- Business object data in an intermediate form, such as a comma-delimited version of a file

The following types of information are not considered user data and are not hidden:

- Business object schemas
- Transaction IDs
- Event IDs
- Call sequences

User authentication

The adapter supports several methods for supplying the user name and password that are needed to connect to the FTP server. By understanding the features and limitations of each method, you can pick a method that provides the appropriate level of security and convenience for your application.

To integrate an adapter into your application, you must provide the user name and password for the adapter to use at run time on WebSphere Process Server or WebSphere Enterprise Service Bus to connect to the FTP server to process outbound requests and inbound events.

At run time, the adapter needs to provide the user name and password to connect to the FTP server. To connect without user intervention, the adapter must access a saved copy of the user information. In a server environment, there are several methods for saving user information. The external service wizard lets you configure the adapter to get the user information using any of the following methods:

- Adapter properties
- Connection specification properties
- J2C authentication alias

Saving the user name and password in adapter properties is a direct way to provide this information at run time. You provide this user name and password when you use the external service wizard to configure your module. Although directly specifying the user name and password seems the most straightforward method, it has important limitations. Adapter properties are not encrypted; the password is stored as clear text in fields that are accessible to others on the server. Also, when the password changes, you must update the password in all instances of the adapter that access that FTP server. This includes the adapters embedded in application EAR files as well as adapters that are separately installed on the server.

Using a data source lets you use a connection already established for another application. For example, if multiple applications access the same database with the same user name and password, the applications can be deployed using the same data source. The user name and password can be known only to the first person who deploys an application to that data source or who defines a data source separately.

Using a J2C authentication data entry, or authentication alias, created with the Java Authentication and Authorization Service (JAAS) feature of Java 2 security is a robust, secure way to deploy applications. An administrator creates the authentication alias that is used by one or more applications that need to access a system. The user name and password can be known only to that administrator, who can change the password in a single place when a change is required.

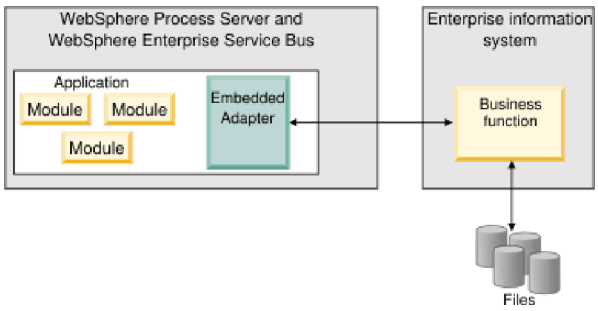
Deployment options

There are two ways to deploy the adapter. You can either embed it as part of the deployed application, or you can deploy it as a stand-alone RAR file. The requirements of your environment affect the type of deployment option you choose.

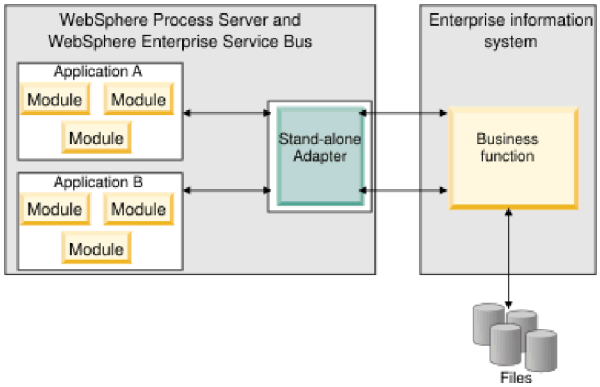
The deployment options are described below:

- **With module for use by single application**: With the adapter files embedded in the module, you can deploy the module to any application server. Use an embedded adapter when you have a single module using the adapter or if multiple modules need to run different versions of the adapter. Using an embedded adapter enables you to upgrade the adapter in a single module without the risk of destabilizing other modules by changing their adapter version.
- **On server for use by multiple applications**: If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.

An embedded adapter is bundled within an enterprise archive (EAR) file and is available only to the application with which it is packaged and deployed.



A stand-alone adapter is represented by a stand-alone resource adapter archive (RAR) file, and when deployed, it is available to all deployed applications in the server instance.



While creating the project for your application using WebSphere Integration Developer, you can choose how to package the adapter [either bundled with the (EAR) file or as a stand-alone (RAR) file]. Your choice affects how the adapter is used in the run time environment, as well as how the properties for the adapter are displayed on the administrative console.

Choosing either to embed an adapter with your application or to deploy the adapter as a stand-alone module depends on how you want to administer the adapter. If you want a single copy of the adapter and do not care about disruption to multiple applications when you upgrade the adapter, then you would be more likely to deploy the adapter as a stand-alone module.

If you plan to run multiple versions, and if you care more about potential disruption when you upgrade the adapter, you would be more likely to embed the adapter with the application. Embedding the adapter with the application allows you to associate an adapter version with an application version and administer it as a single module.

Considerations for embedding an adapter in the application

Consider the following items if you plan to embed the adapter with your application:

- An embedded adapter has class loader isolation.

  A class loader affects the packaging of applications and the behavior of packaged applications deployed on run time environments. *Class loader isolation* means that the adapter cannot load classes from another application or module.

Class loader isolation prevents two similarly named classes in different applications from interfering with each other.

- Each application in which the adapter is embedded must be administered separately.

Considerations for using a stand-alone adapter

Consider the following items if you plan to use a stand-alone adapter:

- Stand-alone adapters have no class loader isolation.

  Because stand-alone adapters have no class loader isolation, only one version of any given Java artifact is run and the version and sequence of that artifact is undetermined. For example, when you use a stand-alone adapter there is only *one* resource adapter version, *one* adapter foundation class (AFC) version, or *one* third-party JAR version. All adapters deployed as stand-alone adapters share a single AFC version, and all instances of a given adapter share the same code version. All adapter instances using a given third-party library must share that library.

- If you update any of these shared artifacts, all applications using the artifacts are affected.

  For instance, if you have an adapter that is working with server version X, and you update the version of the client application to version Y, your original application might stop working.

- Adapter Foundation Classes (AFC) is compatible with previous versions, but the latest AFC version must be in every RAR file that is deployed in a stand-alone manner.

  If more than one copy of any JAR file is in the class path in a stand-alone adapter, the one that is used is random; therefore, they all must be the latest version.

WebSphere Adapters in clustered environments

You can improve adapter performance and availability by deploying the module to a clustered server environment. The module is replicated across all servers in a cluster, regardless of whether you deploy the module using a stand-alone or embedded adapter.

WebSphere Process Server or WebSphere Enterprise Service Bus, WebSphere Application Server Network Deployment, and WebSphere Extended Deployment support clustered environments. Clusters are groups of servers that are managed together to balance workloads and to provide high availability and scalability. When you set up a server cluster, you create a Deployment Manager profile. The HAManager, a subcomponent of the Deployment Manager, notifies the Java 2 Platform, Enterprise Edition (J2EE) Connector Architecture (JCA) container to activate the adapter instance. The JCA container provides a run time environment for adapter instances. For information about creating clustered environments, see the following link: http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm_cluster_v61.html.

Using WebSphere Extended Deployment, you can optionally enhance the performance of adapter instances in your clustered environment. WebSphere Extended Deployment extends the WebSphere Application Server Network Deployment capabilities by using a dynamic workload manager instead of a static workload manager, which is used by WebSphere Application Server Network Deployment. The dynamic workload manager can optimize the performance of

adapter instances in the cluster by dynamically balancing the load of the requests. This means that application server instances can be automatically stopped and started based on the load variations, allowing machines with different capacities and configurations to handle load variations evenly. For information about the benefits of WebSphere Extended Deployment, see http://publib.boulder.ibm.com/infocenter/wxdinfo/v6r1/index.jsp.

In clustered environments, adapter instances can handle both inbound and outbound processes.

Restriction: During inbound communication WebSphere Adapter for FTP is not able to switch polling between a WebSphere Process Server or WebSphere Enterprise Service Bus cluster backup node and the cluster's primary node when each node is installed on a different operating system. For example, if the adapter starts polling on a primary Windows node, it cannot switch to a backup UNIX node because it cannot process the Windows path used for the directory storing in progress events.

High availability for inbound processes

Inbound processes are based on events triggered as a result of updates to data in the FTP server. WebSphere Adapter for FTP is configured to detect updates by polling an event table. The adapter then publishes the event to its endpoint.

Important: In a clustered environment, the event directory must be on a shared file system and not local to any of the cluster machines.

When you deploy a module to a cluster, the Java 2 Platform, Enterprise Edition (J2EE) Connector Architecture (JCA) container checks the enableHASupport resource adapter property. If the value for the enableHASupport property is true, which is the default setting, all of the adapter instances are registered with the HAManager with a policy 1 of N. This policy means that only one of the adapter instances starts polling for events. Although other adapter instances in the cluster are started, they remain dormant with respect to the active event until the active adapter instance finishes processing the event. If the server on which the polling thread was started shuts down for some reason, an adapter instance that is running on one of the backup servers is activated.

Important: Do not change the setting of the enableHASupport property.

High availability for outbound processes

In clustered environments, multiple adapter instances are available to perform outbound process requests. Accordingly, if your environment has multiple applications that interact with WebSphere Adapter for FTP for outbound requests, then you might improve performance by deploying the module to a clustered environment. In a clustered environment, multiple outbound requests can be processed simultaneously, as long as they are not attempting to process the same record.

If multiple outbound requests are attempting to process the same record, such as a Customer address, the workload management capability in WebSphere Application Server Network Deployment distributes the requests among the available adapter instances in the sequence they were received. As a result, these types of outbound requests in a clustered environment are processed in the same manner as those in a single server environment: one adapter instance processes only one outbound

request at a time. For more information about workload management, see the following link: http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm.html.

Adapter customization with Custom Parser Class

The WebSphere Adapter for FTP uses Apache Commons Net API v2.0 to connect to the FTP server. The adapter works with the servers that provide standard listing format such as most UNIX or Linux built-in servers do. If the FTP server ls -l output is different from the standard format, you should make use of the CustomParserClassname property and set an appropriate class name to parse the output. The CustomParserClassName property, which is located in the activation specification and in the managed connection factory, should contain the complete path of the class.

Commons Net API provides an interface, org.apache.commons.net.ftp.FTPFileEntryParser, that you can implement to parse the long list (ls -l) output. By using the class that implements this interface, the adapter can work with FTP servers that do not provide standard listing. The adapter provides a basic implementation of this interface. The class name is com.ibm.j2ca.ftp.util.FTPLongListEntryParser

The following methods are located in the Java™ interface:

```
package org.apache.commons.net.ftp;
public interface FTPFileEntryParser{
FTPFile parseFTPEntry(String listEntry);
String readNextEntry(BufferedReader reader) throws IOException;
List preparse(List original);
}
```

For more information about each of the methods in the Apache Commons Net API v2.0 documentation, see http://commons.apache.org/net/api/index.html

Note: If the FTP server generates MS-DOS type listing (such as the format returned by Windows built-in Internet Information Services (IIS) FTP server configured in MS-DOS directory listing style), you need to implement a class based on the org.commons.apache.net.ftp.parser.NTFTPEntryParser. The NTFTPEntryParser is provided by Apache Commons Net API.

For any other format of the directory listing, implement the appropriate parser class and provide the class name in the Custom Parser Class Name property.

Migrating to version 7.0 of WebSphere Adapter for FTP

By migrating to version 7.0 of WebSphere Adapter for FTP, you automatically upgrade from the previous version of the adapter. Additionally, you can migrate your applications that embed an earlier version of the adapter, so that the applications can use features and capabilities present in version 7.0.

Migration considerations

WebSphere Adapter for FTP version 7.0 may have some features and updates that might affect your existing adapter applications. Before migrating applications that use WebSphere Adapter for FTP, you must consider some factors that might affect your existing applications.

Compatibility with earlier versions

WebSphere Adapter for FTP version 7.0 is fully compatible with the custom business objects (XSD files) and data bindings that are created using the adapter version 6.0.2.x, version 6.1.x, and version 6.2.x and enables the existing business objects and data bindings to work well in the latest version of the adapter.

Because version 7.0 of WebSphere Adapter for FTP is fully compatible with version 6.0.2.x, version 6.1.x, and version 6.2.x, any of your applications that used version 6.0.2.x of WebSphere Adapter for FTP runs unchanged when you upgrade to version 7.0. However, if you want your applications to use features and functionality present in version 7.0 of the adapter, run the migration wizard.

The migration wizard replaces (upgrades) version 6.0.2.x, version 6.1.x or version 6.2.x of the adapter with version 7.0 and enables version 7.0 features and functionality for use with your applications.

Note: The migration wizard does not create components or modify existing components, such as mappers and mediators to work with version 7.0 of the adapters. If any of your applications embed an adapter that is version 6.2.x or earlier and you are upgrading to version 7.0, and you want your applications to take advantage of the features and functions in version 7.0, you might need to change to those applications.

If the artifacts within a module have inconsistent versions, the entire module is marked as unavailable for migration and cannot be selected. Version inconsistencies are recorded in the workspace log, as they indicate that a project might be corrupted.

The adapter migration wizard in WebSphere Integration Developer version 7.0 only supports the migration of adapters from version 6.0.2.x, version 6.1.x, and version 6.2.x to version 7.0. It does not support the adapter migration from the previous versions to version 6.2.x.

Deciding whether to upgrade or to upgrade and migrate

The default processing of the migration wizard is to perform an upgrade of the adapter and to migrate the application artifacts so that the applications can use features and functions in version 7.0 of the adapter. When you choose to upgrade the adapter by selecting a project, the wizard automatically selects the associated artifacts for migration.

If you decide that you want to upgrade the adapter from version 6.0.2.x, version 6.1.x or version 6.2.x to version 7.0, but you do not want to migrate the adapter artifacts, you can do so by deselecting the adapter artifacts from the appropriate area of the migration wizard.

Running the migration wizard without selecting any adapter artifacts installs and upgrades your adapter. As the artifacts are not migrated, your applications cannot take advantage of the features and capabilities that exist in version 7.0 of the adapter.

Migrating multiple adapters referred within a project

When a module contains one or more connector projects, each of which references to different adapters (for example, a module project that contains connector

projects referring to JDBC and SAP adapters), the migration wizard identifies the artifacts belonging to each adapter and migrates these artifacts without disrupting the artifacts of other adapters.

When you select the module project and launch the migration wizard:

- The **Source connector** field lists the connector projects with the selected module project.
- The **Dependent artifact projects** area lists only the selected module project.

If you select the connector project and launch the migration wizard:

- The **Source connector** field lists only the selected connector project.
- The **Dependent artifact projects** area lists all projects which reference the selected connector project, including the module project.

Run the migration wizard in a test environment first

Because adapter migration might require you to change those applications that use version 7.0 of WebSphere Adapter for FTP, you must always perform the migration in a development environment first and test your applications before deploying the application to a production environment.

The migration wizard is fully integrated with the development environment.

Deprecated features

A deprecated feature is one that is supported but no longer recommended and that might become obsolete. The following features from earlier versions of WebSphere Adapter for FTP have been deprecated in version 6.1.x and might require changes to your applications:

- The EventContentType and DefaultObjectName Activation specification properties
- The FTPURL Managed Connection Factory property
- The FTPFileDataBinding data binding
- The annotation tags contained in the XSD files

Performing the migration

You can migrate a project or EAR file to version 7.0 using the adapter migration wizard. When the tool is finished, the migration is complete and you can work in the project or deploy the module.

Before you begin

Review the information in *Migration considerations*.

About this task

To perform the migration in WebSphere Integration Developer, complete the following steps.

Note: After migration is complete, the following changes occur:

- the module will no longer be compatible with previous versions of WebSphere Process Server or WebSphere Enterprise Service Bus, WebSphere Process Server or WebSphere Enterprise Service Bus, or WebSphere Integration Developer.

- an XML data handler is added to all the operations. Because this data handler is not needed for the pass-through operation, you must configure one data binding without the data handler against the pass-through operation.

The following steps describe how to run the adapter migration wizard from the connector project menu while in the Java EE perspective in WebSphere Integration Developer.

Procedure

1. Import the PI (project interchange) file for an existing project into the workspace.

   Note: Ensure that you do not modify the contents of the RAR or copy the adapter jar file outside the connector project.
2. When projects are created in an earlier version of WebSphere Integration Developer, the Workspace Migration wizard starts automatically and selects the projects to migrate. Follow the wizard and complete the workspace migration. For more information, see Migrating workspaces using the WebSphere Integration Developer Migration wizard.
3. Change to the Java EE perspective.
4. Right-click the module and select **Migrate connector project**.

   You can also launch the adapter migration wizard in the following ways:
   - Right-click the project in the Java EE perspective and select **Migrate adapter artifacts**.
   - From the Problems view, right-click a migration-specific message and select **Quick Fix** to correct the problem.

   Note: If the adapter type (for example, CICS/IMS adapter) is not supported by the migration wizard, the **Migrate connector project** and **Migrate adapter artifacts** menus are not available for selection. If the adapter project is of the latest version and the module projects referencing this adapter project are also of the latest version, these menus are disabled.
5. In the Select Projects window, perform the following steps:
   a. The **Source connector** field displays the name of the connector project that you are migrating. If you are migrating a module project, this field lists all the connector projects in the module project. Select the source project from the list. For more information, see Migrating multiple adapters referred within a project
   b. The **Target connector** field displays the name of the connector to which you are migrating. If you are working with more than one adapter version, this list displays the names of all the compatible connectors. Select the connector you want to migrate.
   c. The **Target version** field displays the version corresponding to the target connector that you selected in the previous step.
   d. The **Dependent artifacts project** area lists the adapter artifacts that are migrated. If you are migrating a module project, this area lists only the selected module project. If you are migrating a connector project within the module project, this area lists all projects which reference the selected connector project, including the module project. By default, all the dependent artifact projects are selected. If you do not select a dependent artifact project, that project is not migrated. You can migrate any project that you have not selected at a later time. Previously migrated projects, projects

with a current version, and projects that contain errors are unavailable for migration and are not selected. For more information, see "Upgrading but not migrating a project."

e. Click **Next**. A warning window is displayed with the message, "Properties that are not supported in this version of the target adapter will be removed during the migration"

f. Click **OK**.

6. In the Review Changes window, review the migration changes that occur in each of the artifacts that you are migrating. To view the details, expand each node by clicking the + sign.
7. Click **Finish**.

   Before running the migration process, the wizard performs a backup up of all projects affected by the migration. The projects are backed up to a temporary folder within the workspace. If the migration fails for any reason, or if you decide to cancel the migration before it completes, the wizard deletes the modified projects and replaces them with the projects stored in the temporary folder.

   Upon completing the migration successfully, all backed up projects are deleted.
8. If you are migrating an EAR file, optionally create a new EAR file with the migrated adapter and artifacts, and deploy it to WebSphere Process Server or WebSphere Enterprise Service Bus. For more information about exporting and deploying an EAR file, see the topics devoted to it in this documentation.

   Note: If the adapter module created on version 6.2 uses the FTPS protocol, you need to manually specify the truststore path and truststore password values in the WebSphere Process Server administrative console, after the migration is complete. These values are required to perform server authentication while establishing a connection to SSL.

   Use the key tool utility to import the FTPS server's certificate into the adapter's trust store. For example, enter the command, keytool -import -v -alias serverCert -file server.cert -keystore clientTrustStore, where server.cert is the FTPS server's certificate and clientTrustStore is the trust store of the adapter.

   Set the trust store by updating the JVM property through WebSphere Process Server administrative console. For example, javax.net.ssl.trustStore=C:\MyKeyStore\clientTrustStore, where clientTrustStore is the truststore of the adapter.

   Set the trust store password by updating the JVM property through WebSphere Process Server administrative console. For example, javax.net.ssl.trustStorePassword=truststorepassword

Results

The project or EAR file is migrated to version 7.0. You do not need to run the external service wizard after exiting the adapter migration wizard.

Upgrading but not migrating a project

You can upgrade the adapter from an earlier version, to version 7.0 while choosing not to migrate the adapter project artifacts.

About this task

Running the migration wizard without selecting any adapter artifacts installs and upgrades your adapter. As the artifacts are not migrated, your applications cannot take advantage of the features and capabilities that exist in version 7.0 of the adapter.

Procedure

1. Import the PI (project interchange) file into the workspace.
2. When projects are created in an earlier version of WebSphere Integration Developer, the Workspace Migration wizard starts automatically and selects the projects to migrate. Follow the wizard and complete the workspace migration. For more information, see Migrating workspaces using the WebSphere Integration Developer Migration wizard.
3. In the Java EE perspective, right-click the project name and click **Migrate connector project**. The adapter migration wizard is displayed.
4. In the Select Projects window, clear the dependent artifact projects, and click **Next**. A warning window is displayed with the message, "The properties that are not supported in the version of the target adapter will be removed during the migration."
5. Click **OK**.
6. In the Review Changes window, review the migration changes that occur during updating the project. To view the details, expand each node by clicking the + sign.
7. Click **Finish**.

   Note: When v6.x FTP adapter module configured for FTPS protocol, the truststore of the FTP adapter will be configured in NodeDefaultTrustStore in the **Security -> SSL Certificates and Key Management -> Key stores and certificates** section of Administrative console of the WebSphere Process Server.

   While using v6.x module with FTPS protocol configured, and the adapter is upgraded to version 7.0, the truststore properties needs to be configured in the 'Managed Connection Factory properties' or 'Activation Specification properties'.

   If you prefer to configure the truststore at the administrative console of the WebSphere Process Server, the following additional steps needs to be performed after configuring the truststore in NodeDefaultTrustStore of the WebSphere Process Server Administrative Console.

   a. Go to <java-home>/lib/security/java.security where <java-home> is the directory in which the java file of the WebSphere Process Server is installed. For example, C:\IBM\WebSphere\ProcServer\java\jre\lib\security\java.security
   b. Open the file and find the segment similar as below:

      ```
      # Default JSSE socket factories
      #ssl.SocketFactory.provider=com.ibm.jsse2.SSLSocketFactoryImpl
      #ssl.ServerSocketFactory.provider=com.ibm.jsse2.SSLServerSocketFactoryImpl
      # WebSphere socket factories (in cryptosf.jar)
      ssl.SocketFactory.provider=com.ibm.websphere.ssl.protocol.SSLSocketFactory
      ssl.ServerSocketFactory.provider=com.ibm.websphere.ssl.protocol.SSLServerSocketFactory
      ```

   c. Uncomment the default JSSE socket factories and comment the WebSphere socket factories. The segment will be displayed as below after this setting:

```
# Default JSSE socket factories
ssl.SocketFactory.provider=com.ibm.jsse2.SSLSocketFactoryImpl
ssl.ServerSocketFactory.provider=com.ibm.jsse2.SSLServerSocketFactoryImpl
# WebSphere socket factories (in cryptosf.jar)
#ssl.SocketFactory.provider=com.ibm.websphere.ssl.protocol.SSLSocketFactory
#ssl.ServerSocketFactory.provider=com.ibm.websphere.ssl.protocol.SSLServerSocketFactory
```

d. Restart the WebSphere Process Server.

Results

The project can now be used with WebSphere Adapter for FTP, version 7.0.

Migrating WebSphere Business Integration applications for use with Version 7.0 WebSphere Adapters

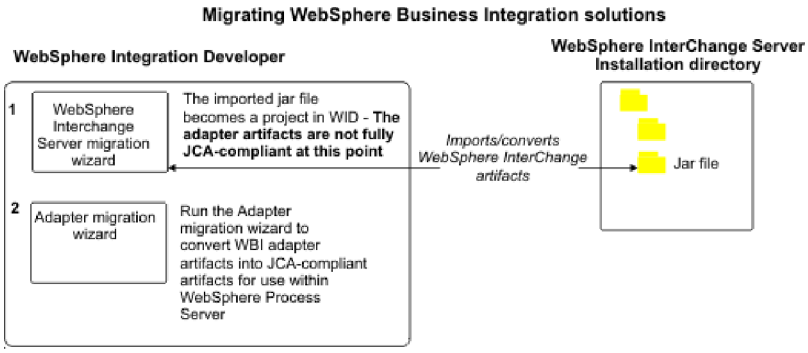
You need to migrate the WebSphere Business Integration applications so that they become compatible with Version 7.0 of your adapter.

About this task

Migrating WebSphere Business Integration applications for use with Version 7.0 of your WebSphere adapter is a multistep process. First, the artifacts from WebSphere InterChange Server are migrated and converted. A project is then created for the artifacts in WebSphere Integration Developer. In the remaining steps, the adapter-specific artifacts are migrated and converted into the JCA-compliant format supported by Version 7.0 of the adapter.

Example

The following diagram shows the wizards that you use to migrate WebSphere Business Integration solutions from WebSphere InterChange Server, so that these applications can be used with Version 7.0 of your adapter.



Roadmap for migrating applications from WebSphere InterChange Server

To use Version 7.0 of WebSphere Adapter for FTP with applications from WebSphere InterChange Server, you need to migrate the application artifacts and convert them so that they can be deployed and run on WebSphere Process Server or WebSphere Enterprise Service Bus. Understanding this task at a high level helps you perform the steps that are needed to accomplish the task.

The following figure illustrates the flow of the migration task. The steps that follow the figure describe this task at a high level only. See the topics following

this roadmap for the details on how to perform each of these steps.

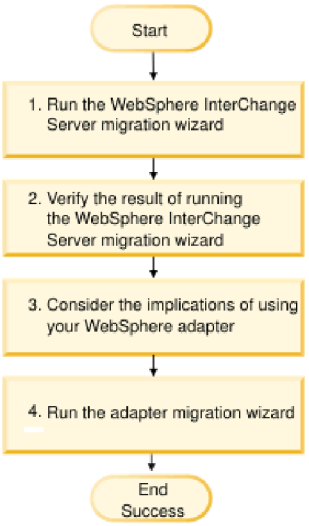


Figure 6. Roadmap for migrating applications from WebSphere InterChange Server

Migrating applications from WebSphere InterChange Server

This task consists of the following steps:

1. Run the WebSphere InterChange Server migration wizard.

   The WebSphere InterChange Server migration wizard moves the application artifacts into WebSphere Integration Developer. The migrated adapter artifacts are not fully JCA-compliant at the completion of this task.
2. Verify that the WebSphere InterChange Server migration is successful.

   Review all messages from the Migration results window and take action if required.
3. Consider the implications of using Version 7.0 of WebSphere Adapter for FTP.

   In addition to considerations for migrating WebSphere InterChange Server applications, you need to consider how Version 7.0 of WebSphere Adapter for FTP works with the migrated applications. Some of the adapter operations supported by WebSphere InterChange Server applications might be supported and implemented differently with Version 7.0 of the adapter.
4. Run the adapter migration wizard.

   Run the adapter migration wizard to update adapter-specific artifacts such as the schemas and service definition files (.import,.export, and .wsdl files) for use with Version 7.0 of the adapter.

Migration considerations for WebSphere Business Integration adapters

By migrating to WebSphere Adapter for FTP Version 7.0, you have an adapter that is compliant with the Java 2 Platform, Enterprise Edition (J2EE) Connector Architecture (JCA) and designed specifically for service-oriented architecture.

Application artifacts

Before running the adapter migration wizard, use the WebSphere InterChange Server migration wizard to generate the application artifacts for the WebSphere Business Integration adapter, including the business objects, maps, and

collaborations. Then you can run the adapter migration wizard to update the adapter-specific artifacts such as the schemas and service definition files (.import,.export, and .wsdl) so that they are suitably converted into a format that is compliant with JCA.

Run the migration wizard in a test environment first

Because migrating from a WebSphere Business Integration adapter to WebSphere Adapter for FTP might require changes to the applications that use Version 7.0 of WebSphere Adapter for FTP, always perform the migration in a development environment first and test your applications before deploying the application to a production environment.

Migrating application artifacts from WebSphere InterChange Server

To migrate the application artifacts into WebSphere Integration Developer, run the WebSphere InterChange Server migration wizard. The wizard imports and converts most of the artifacts into a format that is compatible with WebSphere Process Server or WebSphere Enterprise Service Bus.

Before you begin

Launch the WebSphere InterChange Server migration wizard from within WebSphere Integration Developer to migrate the application artifacts from WebSphere InterChange Server format into artifacts that are compatible with WebSphere Process Server or WebSphere Enterprise Service Bus.

For information about how to prepare to migrate artifacts from WebSphere InterChange Server and for detailed instructions on performing the migration and verifying that the migration was successful, go to the IBM WebSphere Business Process Management information center and read the topic Migrating to WebSphere Process Server from WebSphere InterChange Server.

About this task

Running WebSphere InterChange Server migration wizard might not fully convert adapter-specific artifacts (such as service descriptors, service definitions, and business objects) into WebSphere Process Server or WebSphere Enterprise Service Bus compatible artifacts. To complete the migration of adapter-specific artifacts, run the adapter migration wizard after you have successfully run the WebSphere InterChange Server migration wizard.

Note: While you run the WebSphere InterChange Server migration wizard, ensure that you set each connector in the repository to the same adapter version.

Results

The project and application artifacts are migrated and converted into WebSphere Process Sever compatible artifacts.

What to do next

Run the adapter migration wizard to migrate the adapter-specific artifacts.

Migrating adapter-specific artifacts

After a project is created for the artifacts in WebSphere Integration Developer, you can migrate the project using the adapter migration wizard. The adapter migration wizard updates adapter-specific artifacts such as the schemas and service definition files (.import, .export, and .wsdl) for use with version 7.0 of the adapter. When you finish running the adapter migration wizard, the migration is complete and you can work in the project or deploy the module.

Before you begin

Before running the adapter migration wizard, you should do the following steps:

- Review the information in the "Migration considerations" topic.
- Run the WebSphere InterChange Server migration wizard to migrate the project and convert data objects for use with WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

After migration is complete, the module will work only with Version 7.0 of your adapter.

To perform the migration in WebSphere Integration Developer, complete the following steps.

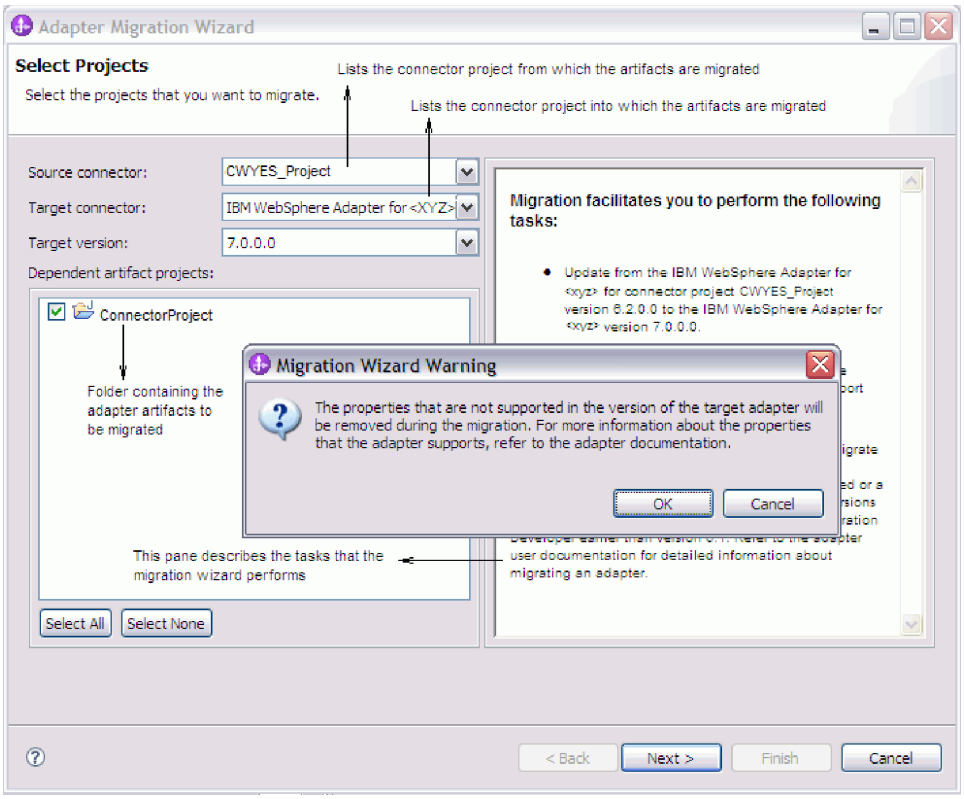
Procedure

1. Import the PI (project interchange) file for an existing project into the workspace.
2. When projects are created in an earlier version of WebSphere Integration Developer, the Workspace Migration wizard starts automatically and selects the projects to migrate. Follow the wizard and complete the workspace migration. For more information, see Migrating workspaces using the WebSphere Integration Developer Migration wizard.
3. Change to the Java EE perspective.
4. Right-click the connector project and select **Migrate connector project**.

   You can also launch the adapter migration wizard by using the right-click option and selecting the module project in the Java EE perspective and selecting **Migrate adapter artifacts**.

   Note:

   If the adapter type (for example, CICS/IMS adapter) is not supported by the migration wizard, the **Migrate connector project** and **Migrate adapter artifacts** menus are not available for selection. If the adapter project is of the latest version and the module projects referencing this adapter project are also of the latest version, these menus are disabled.
   The following figure describes the functional areas of the wizard.



When you launch the migration wizard from the connector project while in the Java EE perspective, by default all the dependent artifact projects are selected. If you do not select a dependent artifact project, that project is not migrated.

5. In the Select Projects window, perform the following steps:
   a. The **Source connector** field displays the name of the connector project that you are migrating. Select the source project from the list.
   b. The **Target connector** field displays the name of the connector to which you are migrating. If you are working with more than one adapter version, this list displays the names of all the compatible connectors. Select the connector to which you want to migrate.
   c. The **Target version** field displays the version corresponding to the target connector you selected in the previous step.
   d. The **Dependent artifacts project** area lists the adapter artifacts that are migrated.
   e. Review the tasks and warnings presented on the welcome page, and click **Next**. A warning window is displayed with the message, "The properties that are not supported in the version of the target adapter are removed during the migration."
   f. Click **OK**.
6. In the Review Changes window, review the migration changes that occur in each of the artifacts that you are migrating. To view the details, expand each node by clicking the + sign.
7. Click **Finish**.

   Before performing the migration process, the wizard backs up all projects affected by the migration. The projects are backed up to a temporary folder within the workspace. If the migration fails for any reason, or if you decide to cancel the migration before it completes, the wizard deletes the modified projects and replaces them with the projects stored in the temporary folder.

8. Select **Project > Clean**, to refresh and rebuild the workspace for the changes to take effect.
9. If you are migrating an EAR file, create a new EAR file with the migrated adapter and artifacts, and deploy it to WebSphere Process Server or WebSphere Enterprise Service Bus. For information about exporting and deploying an EAR file, see "Deploying the module for production" on page 114.

Results

The project is migrated to Version 7.0. You do not need to run the external service wizard after exiting the adapter migration wizard.

Changes to the import, export, and WSDL files after migration

When the WebSphere InterChange Server migration wizard moves the application artifacts into WebSphere Integration Developer, changes made are reflected in the service definition files: the import, export and WSDL files.

The migrated adapter artifacts are not fully JCA-compliant at the completion of this task. You can complete the migration of the adapter-specific artifacts (such as service descriptors, service definitions, and business objects) to a JCA compatible format by running the adapter migration wizard.

Changes to the import file

During migration, the affected module artifacts are migrated to an import file. The existing JMS Binding property is changed to the EIS Binding property in the import file. The other property details added in the import file include information about the data binding configuration, changes to the connection information in the Managed Connection Factory properties, and several new method bindings.

The OutputLog property in WebSphere Business Integration Adapter for JText has the default value Output.log. However, its equivalent property FileSequenceLog in WebSphere Adapter for FTP requires a value that includes the absolute path. For example, C:\Output.log. Hence, after migrating the adapter, you must manually edit the import file for outbound and specify the absolute path for the FileSequenceLog property.

Changes to the export file

During migration, the affected module artifacts are migrated to an export file. The existing JMS Binding property is changed to the EIS Binding property in the export file. The other property details added in the export file include information about the data binding configuration, changes to the connection information in the Activation Specification properties, and several new method bindings.

Changes to the WSDL file after migration

During migration, the affected module artifacts are migrated to corresponding WSDL files that include adapter specific service description WSDL artifacts. The service description files become JCA compatible. The WSDL files will have an input and output type for each operation. Both the inbound and outbound operations work on their specific input types to produce corresponding output types after the operations are performed.

Note:

- When you migrate multiple top level inbound business objects in the project, only the first top-level business object inbound feature works correctly. For the other top level inbound business object to work correctly, you must manually modify the "emit + [verb name] + after image + [business object name]" method in the Input_Processing.java and Input_Async_Processing.java class to call the correct destination services.
- The WebSphere business integration adapter properties that are either not valid or not supported by WebSphere Adapter for FTP are removed from the migrated artifacts.

Chapter 3. Samples and tutorials

To help you use WebSphere Adapters, samples and tutorials are available from the Business Process Management Samples and Tutorials Web site.

You can access the samples and tutorials in either of the following ways:

- From the welcome page of WebSphere Integration Developer, click **Go to Samples and Tutorials**. In the Samples and Tutorials pane, under More samples, click **Retrieve**. Browse the displayed categories to make your selection.
- From the Business Process Management Samples and Tutorials Web site: http://publib.boulder.ibm.com/bpcsamp/index.html.

Chapter 4. Configuring the module for deployment

To configure the adapter so that it can be deployed on WebSphere Process Server or WebSphere Enterprise Service Bus, use WebSphere Integration Developer to create a module, which is exported as an EAR file when you deploy the adapter. You then specify the business objects you want to build and the system on which you want to build them.

Roadmap for configuring the module

Before you use WebSphere Adapter for FTP in a runtime environment, you must configure the module. Understanding this task at a high level helps you perform the steps that are needed to accomplish the task.

You configure the module for WebSphere Adapter for FTP by using WebSphere Integration Developer. The following figure illustrates the flow of the configuration task, and the steps that follow the figure describe this task at a high level only. For the details about how to perform each of these steps, see the topics following this roadmap.

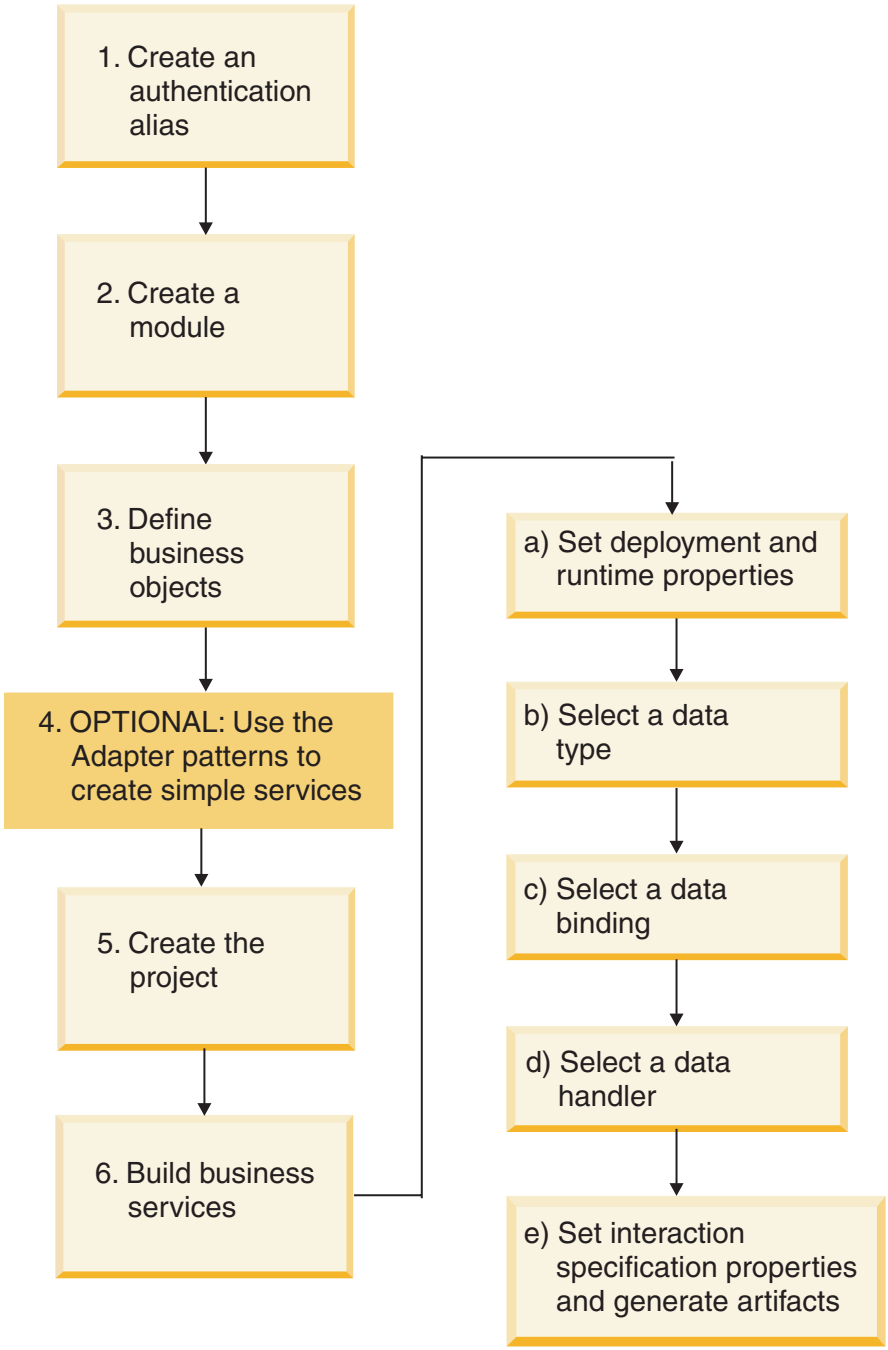


Figure 7. Roadmap for configuring the module

Configuring the module

This task consists of the following steps, which are described at a high level.

Note: These steps assume that you are using user-defined business objects that require data transformation. If using generic business objects, which do not require data transformation, some of the following steps will be ignored. For example, you need not select a data binding and a data handler.

1. Create an authentication alias to access the FTP server. Perform this step using the administrative console on the server.

2. Create a module in WebSphere Integration Developer. You create business objects in the module.
3. Define the business objects that will be used by the project.
4. Use the Adapter patterns wizard to create simple services. For more information, see “Creating a simple service with the adapter pattern wizard” on page 73.
5. Create a project, which is used to organize the files associated with the adapter using the external service wizard in WebSphere Integration Developer.
6. Build business services by running the external service wizard from WebSphere Integration Developer, and then perform the following steps:
   a. Specify the following deployment and runtime properties:
      - Connection properties
      - Security properties
      - Deployment options
      - Function selector - Inbound only
   b. Select a data type and name the operation associated with this data type. For each operation, specify the following:
      - The operation kind. For example, Create, Append, and Exists.
      - Specify if the operation is pass through or user defined.
   c. Select the data binding. Each data type has an equivalent data binding used to read the fields in a business object and fill the corresponding fields in a file.
   d. Select the data handler that performs the conversions between a business object and a native format.
   e. Specify interaction specification property values and generate artifacts. The output from running the external service wizard is saved to a business integration module, which contains the business object or objects, and the import or export file.

Note: If you are performing the step 4, do not follow the other steps following it and exit. If you are not performing the step 4, continue to follow the steps from 5 immediately after the step 3.

Creating an authentication alias

An authentication alias is a feature that encrypts the password used by the adapter to access the FTP server. The adapter can use it to connect to the FTP server instead of using an user ID and a password stored in an adapter property.

Before you begin

To create an authentication alias, you must have access to the administrative console of WebSphere Process Server or WebSphere Enterprise Service Bus. You must also know the user name and password to use to connect to the FTP server.

The following procedure shows you how to gain access to the administrative console through WebSphere Integration Developer. If you are using the administrative console directly (without going through WebSphere Integration Developer), log in to the administrative console and skip to step 2 on page 56.

About this task

Using an authentication alias eliminates the need to store the password in clear text in an adapter configuration property, where it might be visible to ot hers.

To create an authentication alias, use the following procedure.

Procedure

1. Start the administrative console.

   To start the administrative console through WebSphere Integration Developer, perform the following steps:

   a. In the Business Integration perspective of WebSphere Integration Developer, click the **Servers** tab.

   b. If the server does not show the status as **Started**, right-click the name of the server (for example, **WebSphere Process Server**) and click **Start**.

   c. Right-click the name of the server and click **Run administrative console**.

   d. Log on to the administrative console. If your administrative console requires an user ID and a password, type the ID and password and click **Log in**. If the user ID and password are not required, click **Log in**.

2. In the administrative console, click **Security → Secure administration, applications, and infrastructure**.

3. Under **Authentication**, click **Java Authentication and Authorization Service → J2C authentication data**.

4. Create an authentication alias.

   a. In the list of J2C authentication aliases that is displayed, click **New**.

   b. Click the **Configuration** tab, and then type the name of the authentication alias in the **Alias** field.

   c. Type the user ID and password that are required to establish a connection to the FTP server.

   d. Optional: Type a description of the alias.

   e. Click **OK**.

      The newly created alias is displayed.

      The full name of the alias contains the node name and the authentication alias name you specified. For example, if you create an alias on the node widNode with the name ProductionServerAlias, then the full name will be widNode/ProductionServerAlias. This full name is the one you use in subsequent configuration windows.

   f. Click **Save**, and then click **Save** again.

5. Click **New**.

Results

You have created an authentication alias, which you use when you configure the adapter properties.

Creating the module

You create the module in WebSphere Integration Developer. The module allows you to define business objects that will be used by the project.

About this task

Start the external service wizard and follow this procedure to create a new module.

Procedure

1. If WebSphere Integration Developer is not currently running, start it now.
   a. Click **Start** → **Programs** → **IBM WebSphere Integration Developer** → **IBM WebSphere Integration Developer Version 7.0** → **WebSphere Integration Developer Version 7.0**.
   b. If you are prompted to specify a workspace, either accept the default value or select another workspace.

      The workspace is a directory where WebSphere Integration Developer stores your project.
   c. Optional: When the WebSphere Integration Developer window is displayed, click **Go to the Business Integration perspective**.
2. Right-click anywhere within the Business Integration workspace of the WebSphere Integration Developer window, and then select **New** → **Module**.

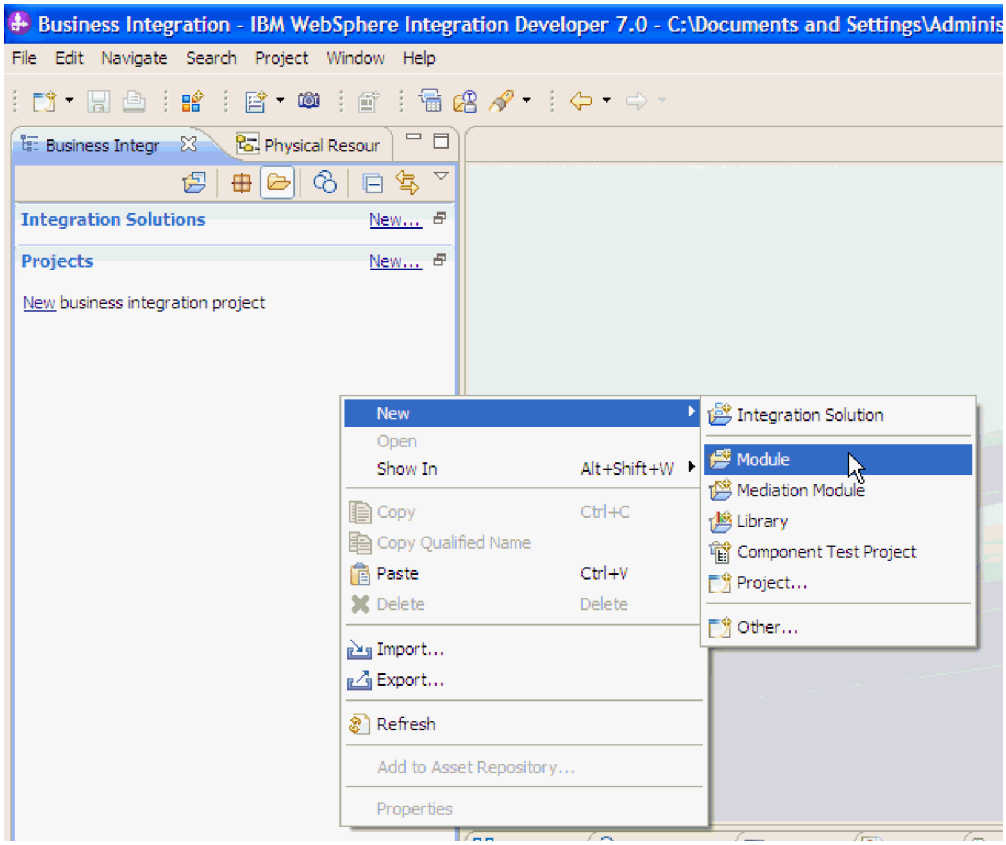


Figure 8. Creating a new module from the Business Integration section of the window

3. Type a name for the field, **Module Name** in the New Module window. For example, FTPOutboundModule. Leave the other options (**Use default location** and **Open module assembly diagram**) checked.
4. Click **Finish**.

Results

A new module is listed in the Business Integration window.

What to do next

Create a project, which is used to organize the files associated with the adapter.

Defining business objects

Predefine the business objects in WebSphere Integration Developer that will be used by the project that you will create in the next topic.

About this task

To predefine new business objects using the business object editor, complete the following steps.

Procedure

1. Expand the new module located inside the Business Integration section of the WebSphere Integration Developer window.
2. Right-click the **Data Types** folder and select **New > Business Object**.
3. Type a new **Name** in the Business Object window. For example, `Customer` to create a customer business object.
4. Click **Finish**. The new business object is added to the **Data Types** folder.
5. Click the **Add a field to a business object** icon and add the necessary fields to the business object.

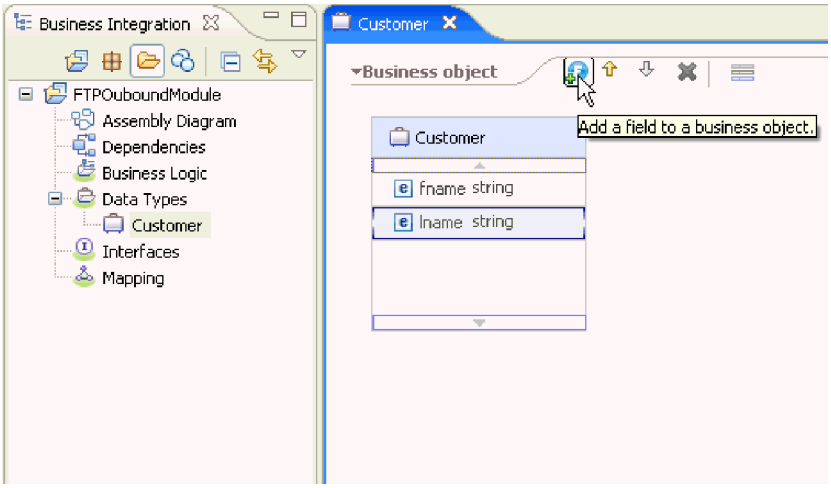


Figure 9. Add Business object fields icon

6. Click the Save icon.
7. Repeat the previous steps for each business object that you want to create.

Results

The new business objects are defined.

What to do next

Create a project, which is used to organize the files associated with the adapter.

Converting business objects to COBOL copybook files during outbound processing

Use the external service wizard in WebSphere Integration Developer to generate business object definitions from a COBOL program source file. These business object definitions are used during outbound processing.

Before you begin

Before you perform this task, make sure that:

1. You have created a module in WebSphere Integration Developer.
2. The COBOL program source file (.ccp file) is in a local directory on your workstation.
3. If you are going to generate a wrapper business object definition, you have imported the adapter RAR file into your workspace.

About this task

Use the external service wizard to generate a business object definition for a COBOL program source file. After you have generated the business object definition, you can run the external service wizard again to generate a wrapper business object definition from the generated business object.

Procedure

1. To generate the business object for the COBOL program source file:
   a. In the Business Integration section of the window, right-click the module and select **New → Business Object From External Service**.
   b. In the Input Sources For Creating Business Objects window, expand **Languages** and select **Cobol**.
   c. Click **Next**.

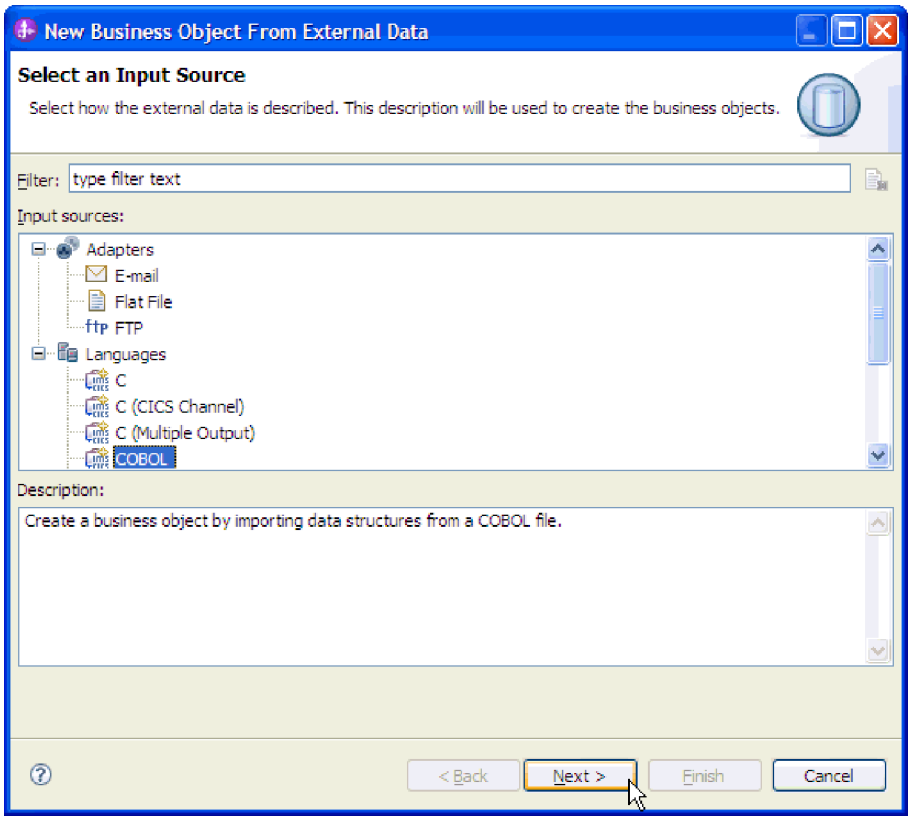


Figure 10. Select an Input Source window

d. In the Business Object Mapping Details window, make sure the **Selected mapping** value is **COBOL to Business Object**. Click **Browse** and select the .ccp file. For example, `taderc99.ccp` can be the name of the .ccp file.

e. Click **Next**.

f. In the Select Data Structures window, click **Find**. The new business object called DFHCOMMAREA is displayed.

g. Select DFHCOMMAREA and click **Next**.

h. Click **Finish**.

A business object called DFHCOMMAREA is created in the module.

2. Optional: Generate a wrapper business object definition. Wrapper business object definitions wrap existing business object definitions with additional function. The generation of wrapper business object definitions is optional. The option to generate wrapper business object definitions is displayed only if the adapter RAR file is imported into the workspace.

   Note: If you want to generate wrapper business object definitions, then you must generate them before running the external service wizard.

   a. In the Business Integration section of the window, right-click the module and select **New → Business Object From External Service**.

   b. In the Input Sources For Creating Business Objects window, expand **Adapters**, select the adapter connector project for which you want to generate the wrapper business object. In this case, select FTP.

   c. Click **Next**.

   d. In the Select an Adapter window, select Adapter for FTP (IBM: 7.0.0.0) adapter, and click the CWYFT_FTPFile connector project. Click **Next**.

e. In the Business Object Properties window, click **Browse** and select the business object created in Step 1, for example, DFHCOMMAREA, for the data type.

f. To generate a business graph, select the **Generate business graph for each business object** check box. For generating the retrieve wrapper, select the **Generate retrieve container to retrieve multiple business objects** check box.

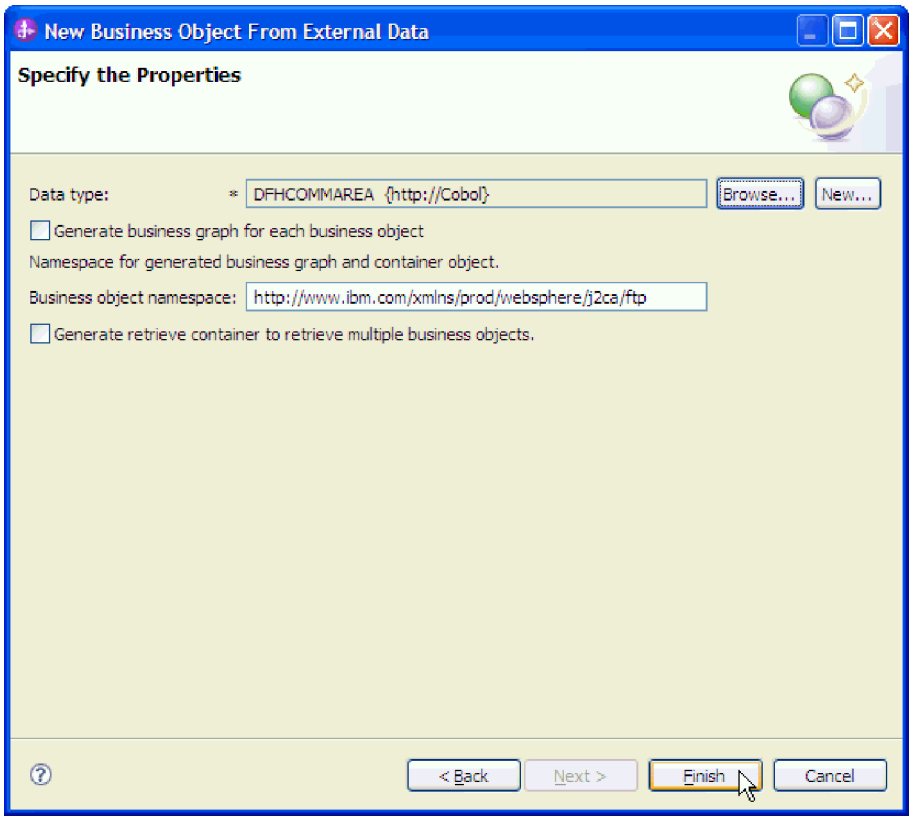


Figure 11. Specify the Properties window

g. Click **Finish**.

A wrapper business object and a business graph, called DFHCOMMAREAWrapper and DFHCOMMAREAWrapperBG, respectively, are listed for the current module in the Business Integration window. If the user has selected **Generate business graph for each business object** and **Generate retrieve container to retrieve multiple business objects**, then a business object called DFHCOMMAREARetrieveWrapper and a business graph called DFHCOMMAREARetrieveWrapperBG are also listed for the current module in the Business Integration window.

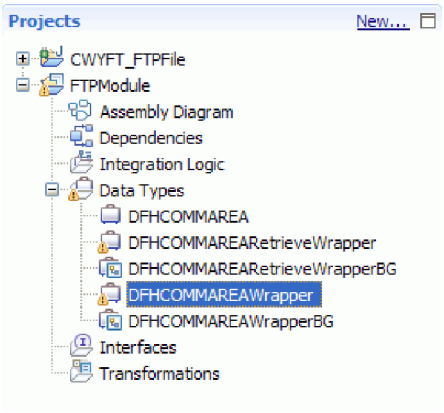


Figure 12. The wrapper business object and the business graph listed in the Business Integration window

3. Generate the required artifacts for the COBOL copybook outbound module. This example shows the configuration for a Create operation.
   a. In the Business Integration section of the window, right-click on the module and select **New** → **External Service**.
   b. Select **FTP** from **Adapters** and click **Next**.
   c. In the Select an Adapter window, select **IBM WebSphere Adapter for FTP (IBM : 7.0.0.0)** adapter, and click the **CWYFT_FTPFile** connector project. Click **Next**.
   d. In the Processing Direction window, select **Outbound**.
   e. Click **Next**.
   f. From the Service Configuration Properties window, in the **Data format options** list, select **Use COBOL, C or PL/I data binding**.

      Note: This is not a data binding, but a data binding generator. The tool generates the appropriate data binding code for you in the current module.

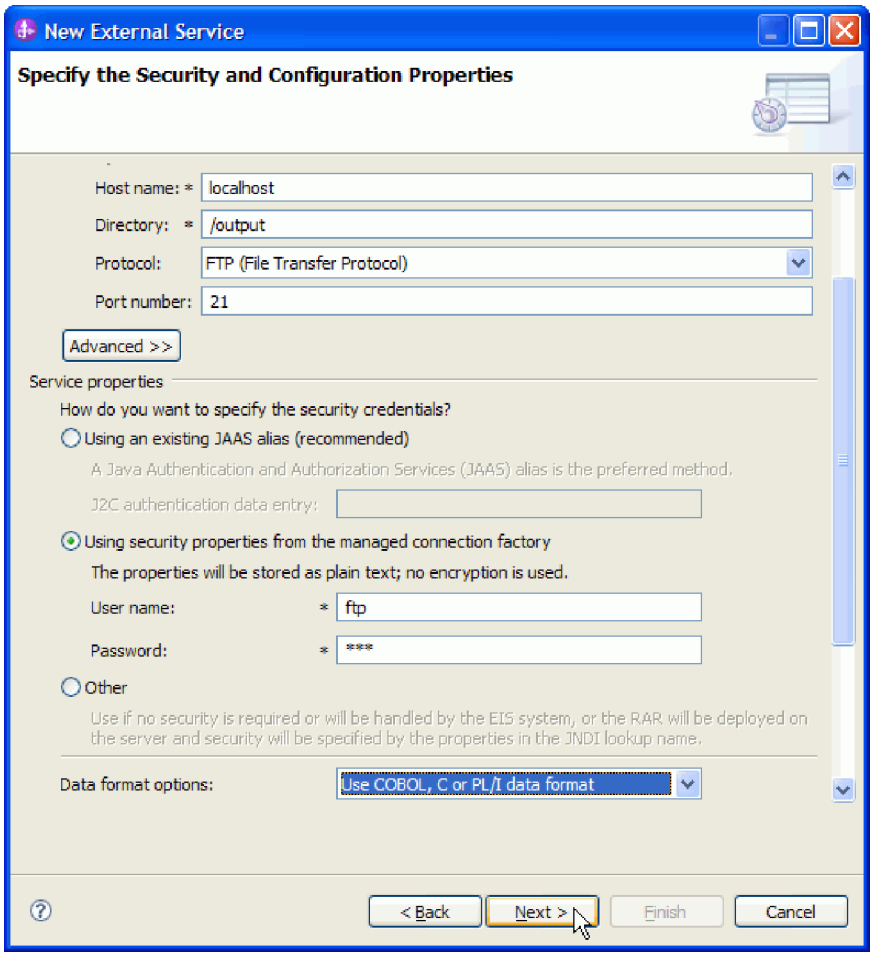


Figure 13. Specify the Security and Configuration Properties window

g. Specify the other required properties for the outbound operation, and click **Next**.

h. In the Operations window, click **Add** and then **Create**. For the retrieve operation, select **Retrieve**. Select the data type as **User defined type** from the list and click **Next**.

i. Browse for the input type (either DFHCOMMAREA, DFHCOMMAREAWrapper, or DFHCOMMAREAWrapperBG) and click **OK**. For the **Retrieve** operation, browse for the appropriate output type (either DFHCOMMAREA, DFHCOMMAREARetrieveWrapper, or DFHCOMMAREARetrieveWrapperBG).

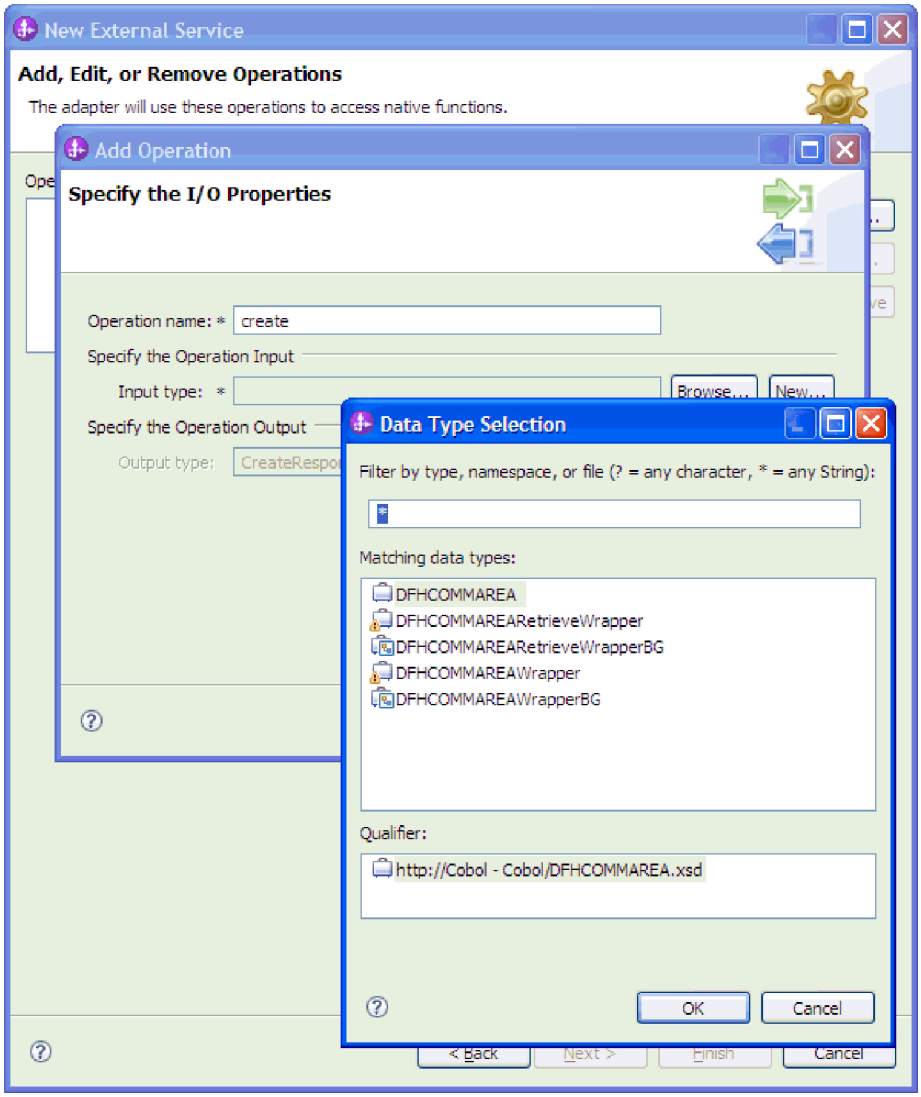


Figure 14. The Data Type Selection window

j. Click **Next**.

The data bindings used by the COBOL copybook, WSDL files, import files, and other artifacts are generated. See the Project Explorer window for the generated data binding classes.

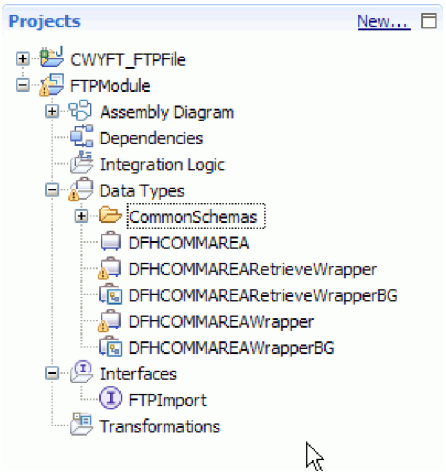


Figure 15. Data bindings used by the COBOL copybook, WSDL files, import files, and other artifacts

Results

A business object, a wrapper business object, and a business graph are created for the COBOL program source file for the outbound module. Artifacts are generated for an outbound Create operation that uses the COBOL copybook data binding. This module can be deployed on WebSphere Process Server and tested for the Create operation.

Note: To generate artifacts for other supported operations (Append and Overwrite), follow the same steps, beginning with Step 3(h).

What to do next

Deploy the module.

Converting COBOL copybook files to business objects during inbound processing

Use the external data wizard in WebSphere Integration Developer to generate business object definitions from a COBOL program source file. These business object definitions are used during inbound processing.

Before you begin

Before you perform this task, make sure that:

1. You have created a module in WebSphere Integration Developer.
2. The COBOL program source file (.ccp file) is in a local directory on your workstation.
3. You have created a local event directory.
4. If you are going to generate a wrapper business object definition, you should have imported the adapter RAR file into your workspace.

About this task

Use the external data wizard to generate a business object definition for a COBOL program source file. After you have generated the business object definition, you

can optionally rerun the external data wizard to generate a wrapper business object definition from the generated business object.

Procedure

1. Generate the business object definition for the COBOL program source file.
   a. In the Business Integration section of the window, right-click the module and select **New → Business Object From External Service**.
   b. In the Input Sources For Creating Business Objects window, expand **Languages** and select **COBOL**.
   c. Click **Next**.

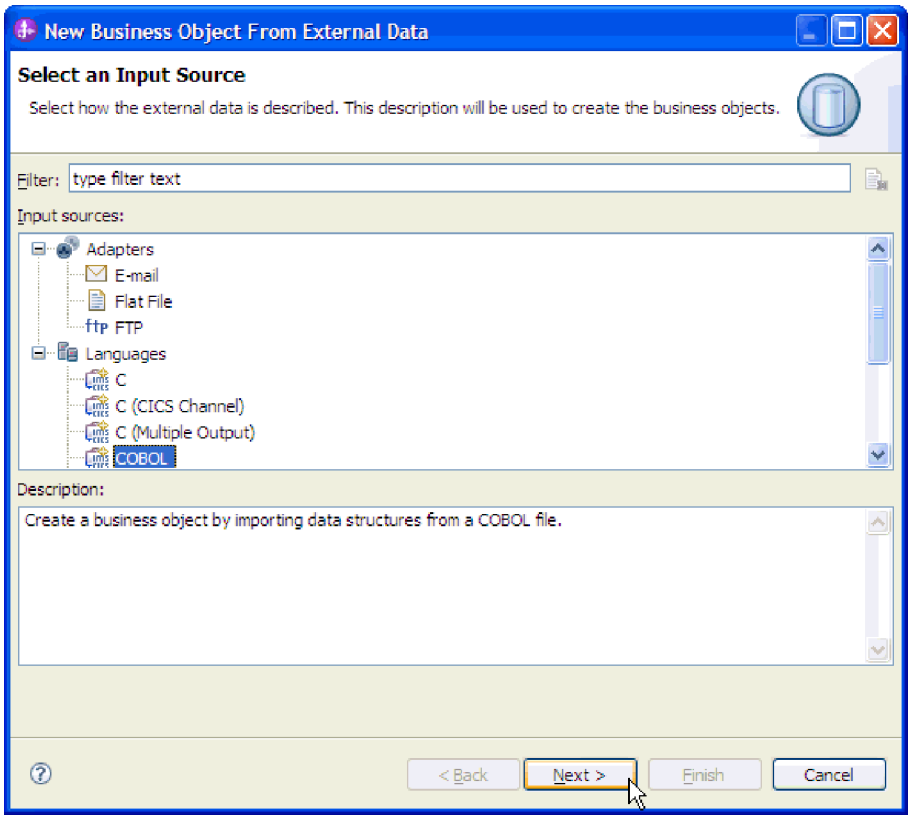


Figure 16. Select an Input Source window

   d. In the Business Object Mapping Details window, make sure the **Selected mapping** value is **COBOL to Business Object**. Click **Browse** and select the .ccp file. For example, `taderc99.ccp` can be the name of the .ccp file.
   e. Click **Next**.
   f. In the Select Data Structures window, click **Find**. The new business object, called DFHCOMMAREA is displayed.
   g. Select DFHCOMMAREA and click **Next**.
   h. Click **Finish**.

   A business object, called DFHCOMMAREA is created in the module.
2. Optional: Generate a wrapper business object definition. Wrapper business object definitions wrap existing business object definitions with additional function. The generation of wrapper business object definition is optional. The option to generate wrapper business object definitions are displayed only if the adapter RAR file is imported into the workspace.

Note: If you want to generate wrapper business object definitions, then you should generate them before running the external service wizard.

a. In the Business Integration section of the window, right-click the module and select **New** → **Business Object From External Service**.

b. In the Input Sources For Creating Business Objects window, expand **Adapters**, select the adapter connector project for which you want to generate the wrapper business object. In this case, select FTP.

c. Click **Next**.

d. In the Select an Adapter window, select the connector project in which the new business object was saved and click **Next**.

e. In the Business Object Properties window, click **Browse** and select the business object created in Step 1, for example, DFHCOMMAREA, for the data type.

f. To generate a business graph, select the **Generate business graph for each business object** check box.

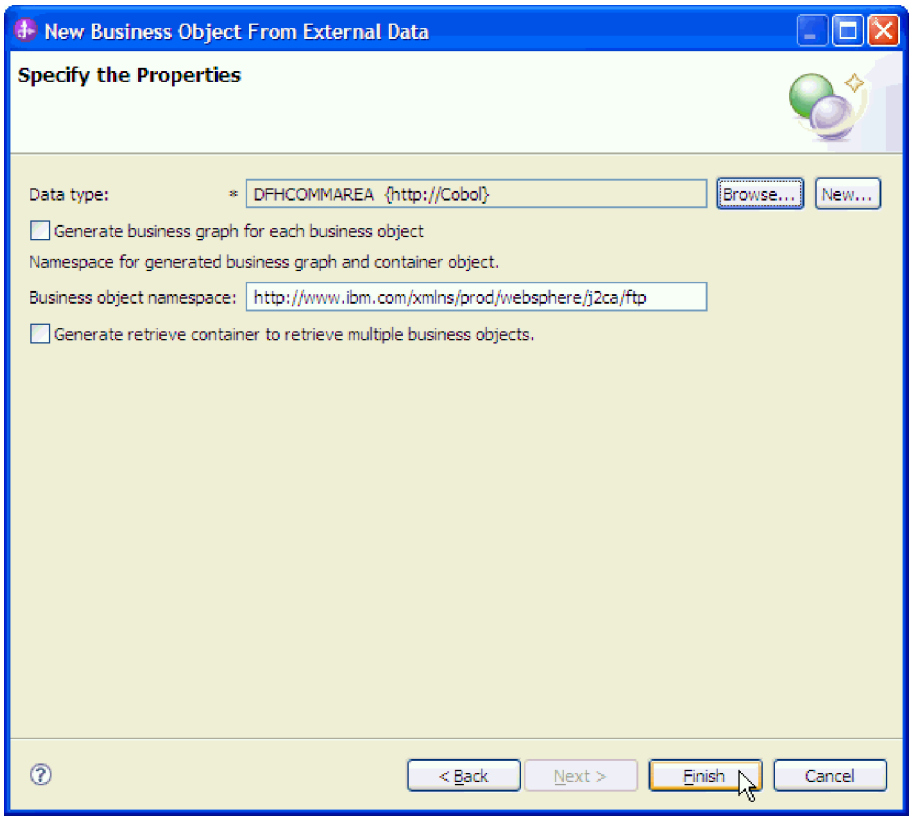


Figure 17. Specify the Properties window

Note: Generate retrieve container to retrieve multiple business objects is applicable only for outbound retrieve operation.

g. Click **Finish**.

A wrapper business object and a business graph, called DFHCOMMAREAWrapper and DFHCOMMAREAWrapperBG, respectively are listed for the current module in the Business Integration window.

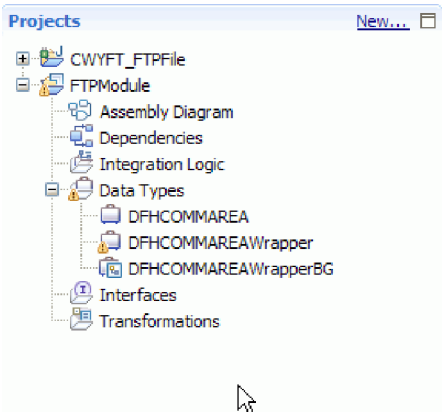


Figure 18. The wrapper business object and the business graph listed in the Business Integration window

3. Generate the required artifacts for the COBOL copybook inbound module.
   a. In the Business Integration section of the window, right-click the module and select **New → External Service**.
   b. Select **FTP** under **Adapters** and click **Next**.
   c. In the Select an Adapter window, select **IBM WebSphere Adapter for FTP (IBM : 7.0.0.0)** adapter, and click the **CWYFT_FTPFile** connector project. Click **Next**.
   d. In the Processing Direction window, select **Inbound** and click **Next**.
   e. Click **Browse** and select the event directory.
   f. For the **Function selector**, select the default value.
   g. In the **Data format options** list, select **Use COBOL, C or PL/I data binding option**.

      Note: This is not a data binding, but a data binding generator. The tool generates the appropriate data binding code in the current module.

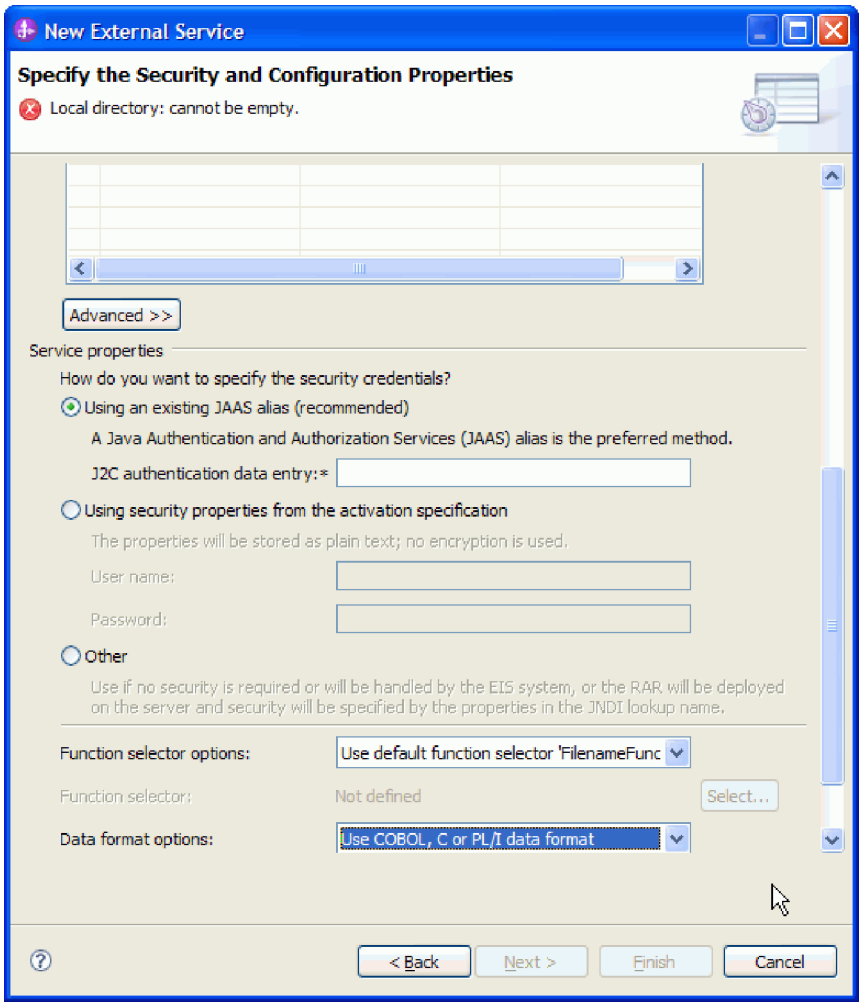


Figure 19. Specify the Security and Configuration Properties window

h. Optional: If the input file contains multiple COBOL program source files, you can enable file splitting by specifying a size or delimiter. To enable file splitting, click **Advanced** and then click **Additional configuration**. To enable file splitting by size, you must provide the correct length of each COBOL program source file. You can either open the business object in a text editor and add the maximum length, or view the content size of DFHCOMMAREA at the top of the file. See "Specify criteria to split file content property (SplitCriteria)" on page 216.

i. Click **Next**.

j. In the Operations window, click **Add**.

k. In the Operations window, select **User defined type** as the data type. Click **Next**.

l. For the input type, click **Browse** and select the generated business object (DFHCOMMAREA). Click **OK**.

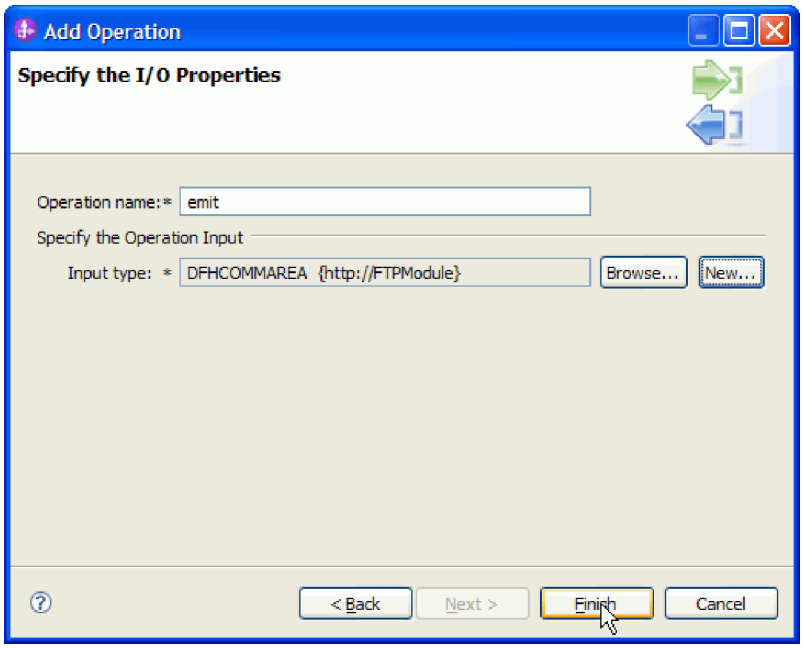


Figure 20. Specify the I/O Properties window

m. Click **Finish**.

n. Click **Next** and then **Finish**.

The data bindings used by COBOL copybook, WSDL files, export files, and other artifacts are generated. See the Project Explorer window for the generated data binding classes.

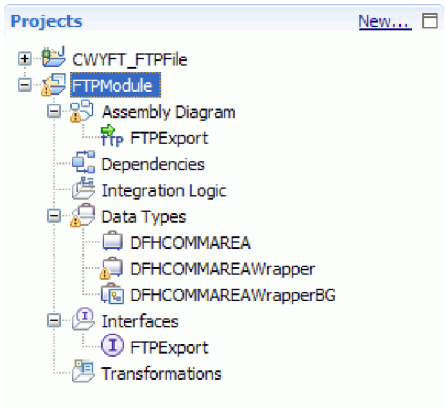


Figure 21. Data bindings used by COBOL copybook, WSDL files, export files, and other artifacts

Results

A business object, a wrapper business object, and a business graph are created for the COBOL program source file for the inbound module. Artifacts are generated for an inbound operation that uses COBOL copybook data binding. This module can be deployed on WebSphere Process Server and tested for an inbound operation.

What to do next

Deploy the module.

Defining WebSphere Application Server environment variables

Use the administrative console of the runtime environment to define WebSphere Application Server environment variables.

About this task

To define a WebSphere Application Server environment variable, use the following procedure.

Procedure

1. Start the administrative console of the server.
2. From the left menu, select **Environment** → **WebSphere Variables**.
3. Select the scope for the environment variable. The scope specifies the level at which the resource definition is visible on the administrative console panel. The possible values are server, node, and cell. In this example, we are choosing `Cell=widCell`.

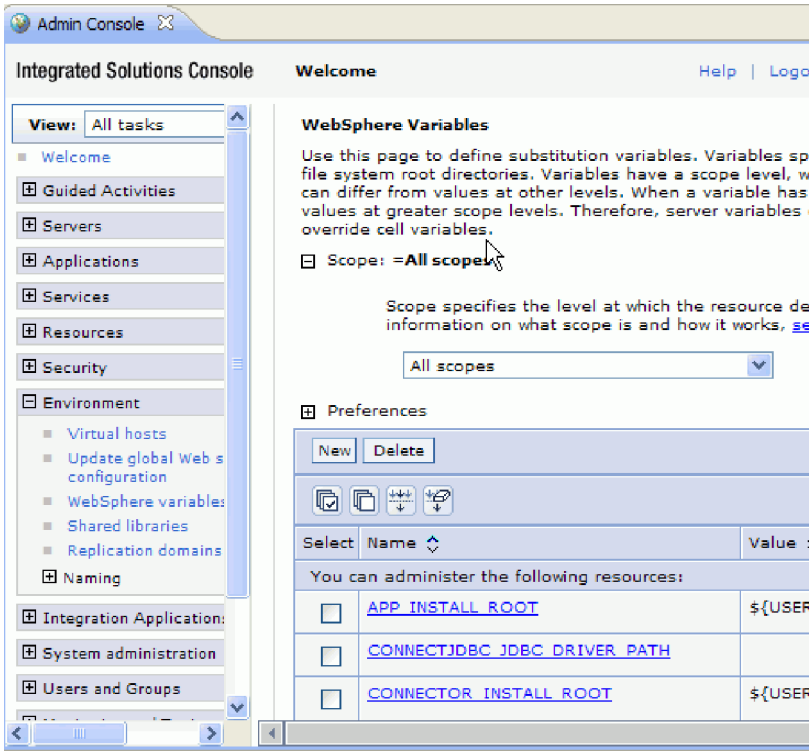


Figure 22. Setting the scope for the environment variable

4. Click **New**, and provide a name and a value for the environment variable. The name is the symbolic name that represents a physical path. The value is the absolute path that the variable represents. In this example, the name is `EVENT_DIRECTORY` and the value is `/home/user/event`. You can use the **Description** field, which is optional, to describe the purpose of the variable.

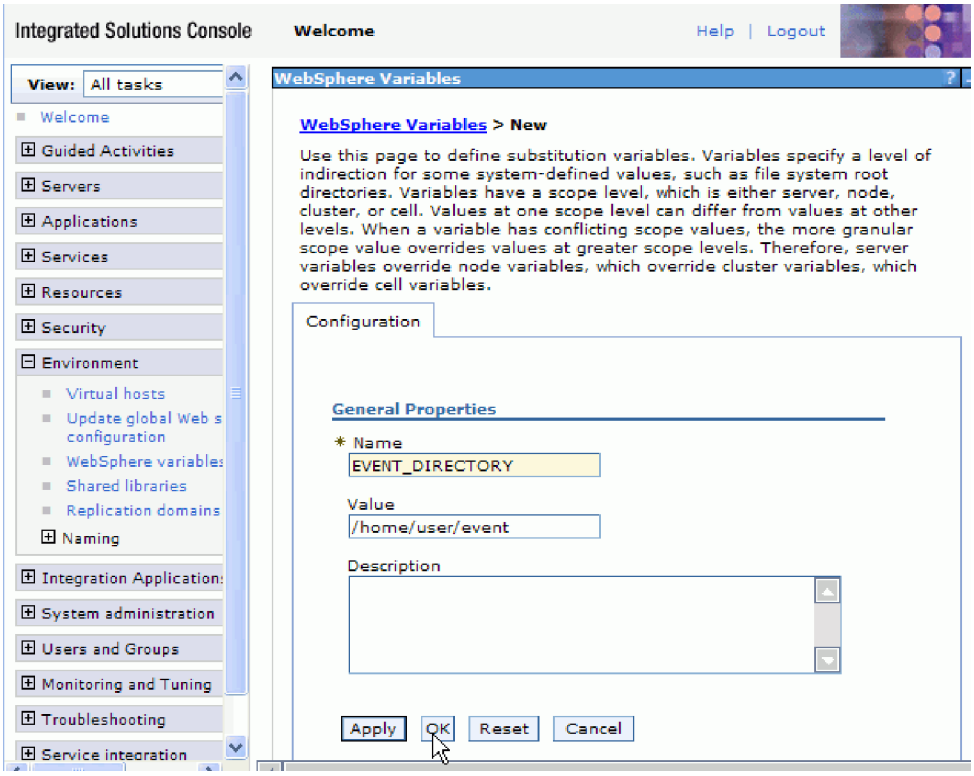


Figure 23. Providing a name and a value for the environment variable

5. Click **OK** and save the changes.

Results

An environment variable called `EVENT_DIRECTORY` is created, with the value `/home/user/event` and a scope of `Cell=widCell`. You can now use it in the external service wizard whenever you need to specify the event directory.

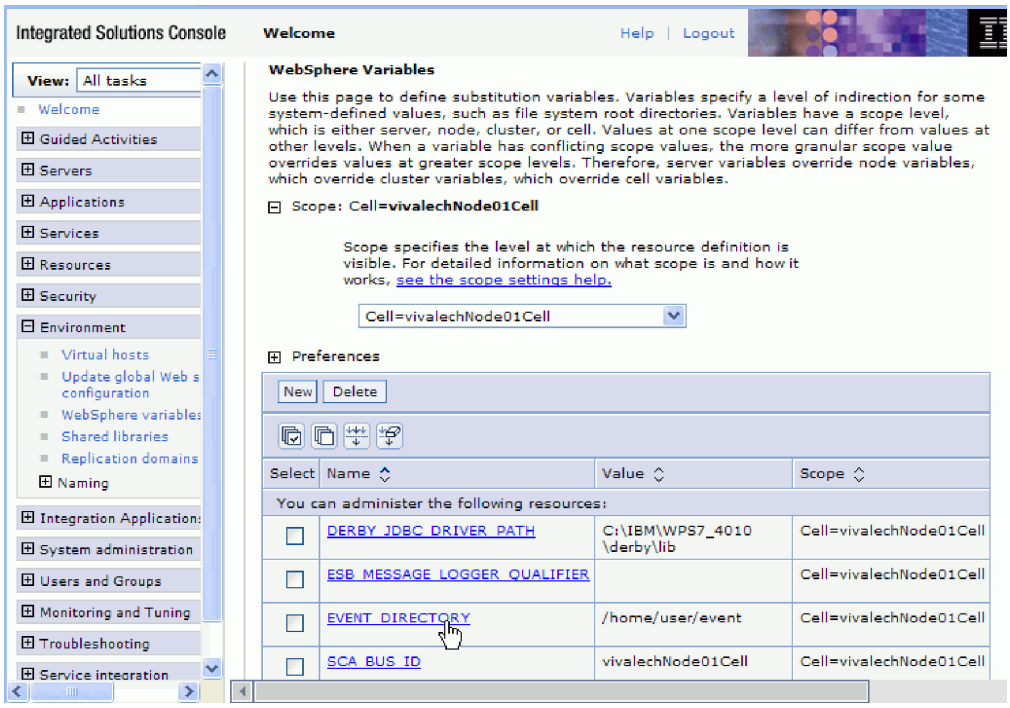


Figure 24. The new environment variable EVENT_DIRECTORY displayed in the WebSphere Variables window

What to do next

Create a project, which is used to organize the files associated with the adapter.

Creating a simple service with the adapter pattern wizard

Adapter patterns provide a quick and easy way of creating a simple service with an adapter.

Before you begin

A module has already been created called RetrieveAFileModule and a business object called Customer has already been created. If you are using WebSphere Application Server environment variables to specify local files and directories, you have defined them using the WebSphere Process Server administrative console.

About this task

The following adapter patterns are available for the adapter for FTP:

Table 6. Adapter pattern details

Adapter pattern	Description
Inbound FTP pattern	The FTP inbound pattern creates a service that retrieves a file in a specific directory on an FTP server. If the file is not in an XML format, you can specify a data handler that will transform from the file content format to business objects. The file content can be split if the content contains multiple copies of the data structure for processing.

Table 6. Adapter pattern details (continued)

Adapter pattern	Description
Outbound FTP pattern	The FTP outbound pattern creates a service that stores data in a file in a specific directory on an FTP server. If the required output format is not an XML format, you can specify a data handler that will transform the business object to the file content format.

In this example, you create an FTP inbound service that receives a file from the file system for processing. The completed service in this example will read in a file and split the contents into separate files based on a delimiter.

Complete the following steps to create a service with the adapter pattern wizard:

Procedure

1. Open the Assembly diagram of RetrieveAFileModule
2. Expand **Inbound Adapters**, drag and drop FTP into the Assembly diagram.
3. Select **Simple:Create an inbound FTP service to read from a remote file**.
4. Click **Next**.

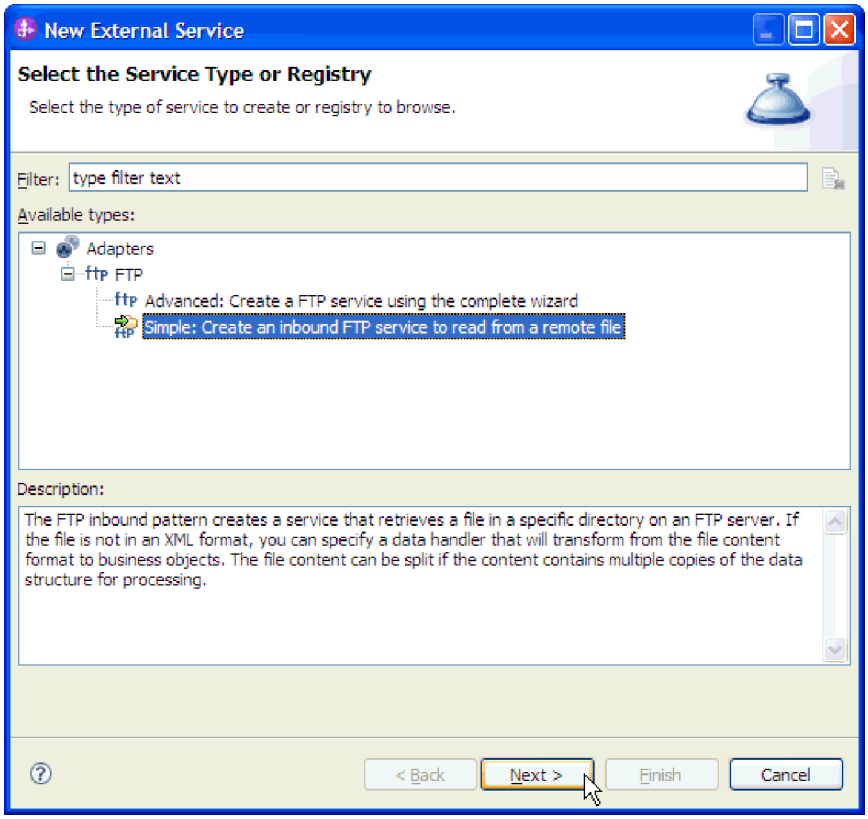


Figure 25. Select the Service Type or Registry window

5. In the FTP service name window, specify the name, something meaningful such as `FTPInboundInterface` and click **Next**.
6. In the Business object and location window, click **Browse** and navigate to the **Customer** business object.

7. Specify the directory where you placed the input file, in this case the `/home/user/event` directory, and click **Next**. To use a WebSphere Application Server environment variable for this value, specify the name of the variable in braces, preceded by a `$` symbol. For example: `${FTPINBOUNDEVENTS}`.

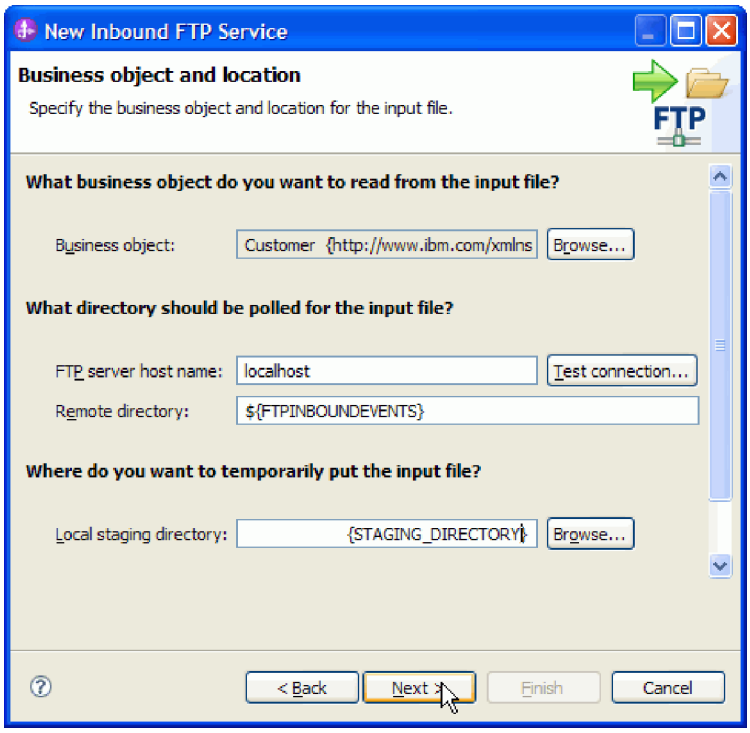


Figure 26. Business object and location window

8. In the FTP server security credential window, select either **Using an existing JAAS alias** or **Using user name and password** and click **Next**.
9. In the Input file format and file content split option window, accept the default XML input file format or select **Other** and specify a data handler to transform the data from your native format to the business object format.
10. Select **Split file content by delimiter** and enter your delimiter, which is `####;\n` in this example. Click **Next**.

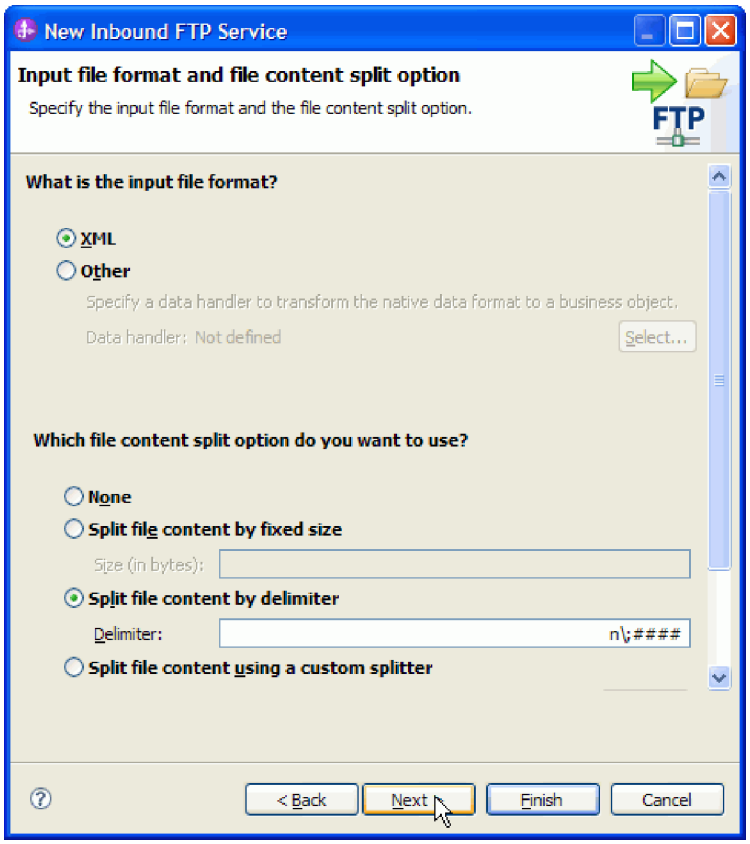


Figure 27. Input file format and file content split option window

11. In the Archive directory and wrapper business object window, specify the **Local archive directory**, which is `FTP\inboundarchive` in this example. To use a WebSphere Application Server environment variable for this value, specify the name of the variable in braces, preceded by a $ symbol. For example: `${FTPINBOUNDARCHIVE}`. Select **Use a wrapper business object to contain additional input file information** check box, if you want to include the adapter-specific information. Click **Finish**.

Results

The inbound service is created, which includes the following artifacts:

Table 7. Artifact details

Artifact	Name	Description
Export	FTPInboundInterface	The export exposes the module externally, in this case, to the WebSphere Adapter for FTP.
Business objects	Customer, CustomerWrapper	The Customer business object contains the fields for customer data such as name, address, city, and state. The CustomerWrapper business object contains additional fields for adapter-specific information.

Table 7. Artifact details (continued)

Artifact	Name	Description
Interface	FTPInboundInterface	This interface contains the operation that can be invoked.
Operation	emitCustomerInput	emitCustomerInput is the only operation in the interface.

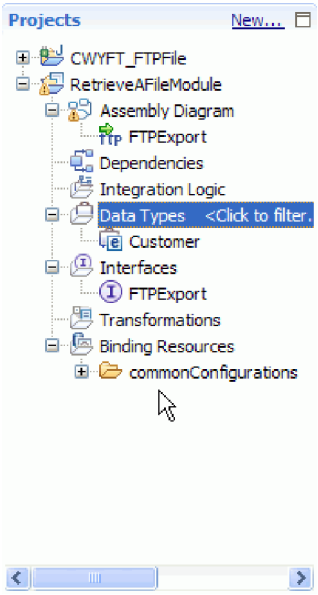


Figure 28. The **Business Integration** *section of the WebSphere Integration Developer window with the new artifacts*

Starting the external service wizard

To begin the process of creating and deploying a module, you start the external service wizard in WebSphere Integration Developer. The wizard creates a project that is used to organize the files associated with the module.

Before you begin

Make sure that you have gathered the information you need to establish a connection to the FTP server. For example, you need the name or IP address of the FTP server and the user ID and password needed to access it.

About this task

Start the external service wizard to create a project for the adapter in WebSphere Integration Developer. If you have an existing project, you can select it instead of having the wizard create one.

To start the external service wizard and create a project, use the following procedure.

Procedure

1. To start the external service wizard, go to the Business Integration perspective of WebSphere Integration Developer, and then click **File → New → External Service**.
2. In the New External Service window, expand **Adapters**.
3. Expand FTP in **Adapters** and select **Advanced: Create a FTP service using the complete wizard** and click **Next**.
4. In the Select an Adapter window, select the adapter name to create a new project, or select an existing project to reuse it.
   - To create a project, perform the following steps:
     a. Select **IBM WebSphere Adapter for FTP (IBM : 7.0.0.0)** and click the CWYFT_FTPFile connector project. Click**Next**.
     b. In the Adapter Import window, provide details about the project you want to create.
        1) In the **Connector project** field, optionally specify a different name for the project.
        2) In the **Target runtime** field, select the server (for example, **WebSphere Process Server v7.0**).
        3) Click **Next**.
   - To select an existing project, select the project folder under **IBM WebSphere Adapter for FTP (IBM : 7.0.0.0)** and then click **Next**.

Results

For a new project, the project is created and is listed in the Business Integration perspective. The wizard will create adapter artifacts in the specified project.

Configuring the module for outbound processing

To configure a module to use the adapter for outbound processing, use the external service wizard in WebSphere Integration Developer to build business services, specify data transformation processing, and generate business object definitions and related artifacts.

Setting deployment and runtime properties

Specify deployment and runtime properties that the external service wizard uses to connect to the FTP server.

Before you begin

Before you can set the properties in this section, you must have created your adapter module. It should be displayed in WebSphere Integration Developer below the adapter project. For more information about creating the adapter project, refer to “Starting the external service wizard” on page 77 topic.

About this task

To set deployment and runtime properties, follow this procedure. For more information about the properties in this topic, refer to “Managed (J2C) connection factory properties” on page 158 topic.

Procedure

1. In the Processing Direction window, select **Outbound** and click **Next**.
2. In the **Deploy connector project** field, specify whether to include the adapter files in the module. Choose one of the following options:
   - **With module for use by single application**

     With the adapter files embedded in the module, you can deploy module to any application server. Use an embedded adapter when you have a single module using the adapter or when multiple modules need to run different versions of the adapter. By using an embedded adapter, you can upgrade the adapter in a single module without the risk of destabilizing other modules by changing their adapter version.
   - **On server for use by multiple applications**

     If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.
3. Define the following FTP system connection information for your module. For more information, refer to "Managed (J2C) connection factory properties" on page 158 topic.
   - **Host name** - Specifies the host name of the FTP server.
   - **Directory** - Specifies the output directory on the FTP server.
   - **Protocol** - Specifies the protocol used to connect to the FTP server. Following are the protocols that can be specified:
     - FTP - File Transfer Protocol
     - FTP over SSL - File Transfer Protocol over Secure Socket Layer
     - FTP over TLS - File Transfer Protocol over Transport Layer Security
     - SFTP - Secure shell File Transfer Protocol
   - **Port number** - Specifies the port number of the FTP server.

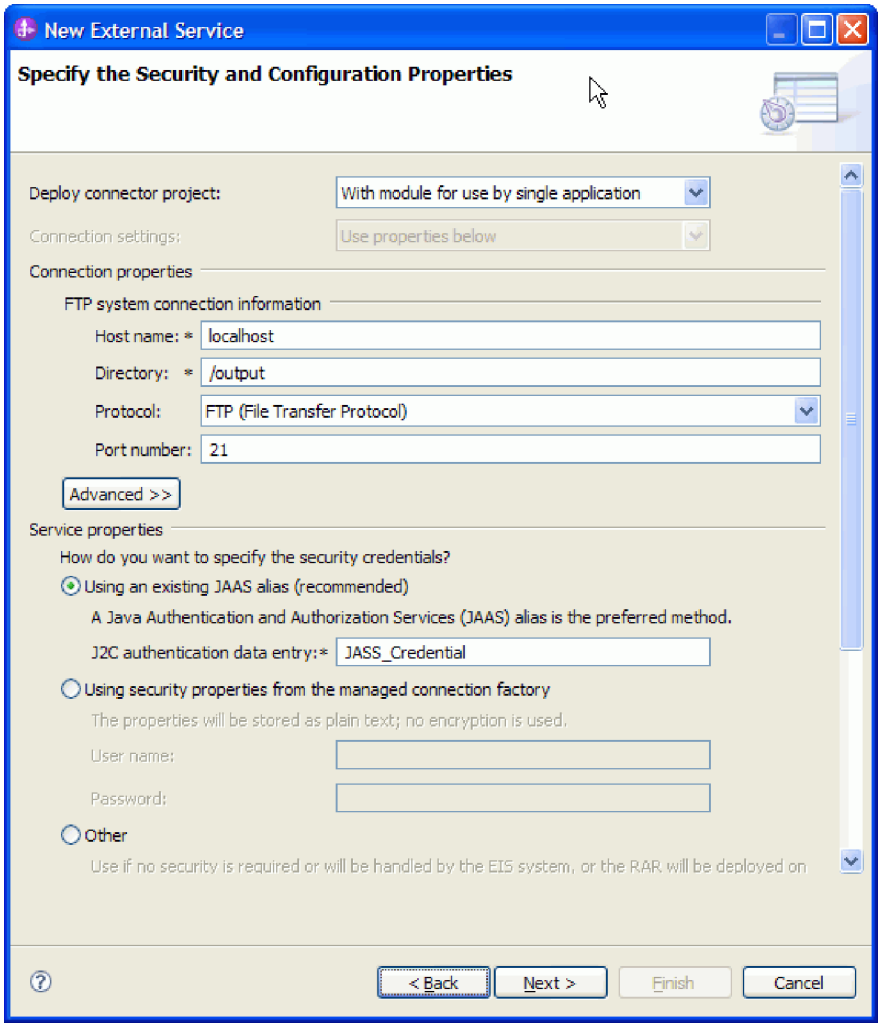


Figure 29. Specify the Security and Configuration Properties window

4. Click **Advanced** to specify additional properties, service properties, data format options, those that control working with a second FTP server, bidi formatting, a staging directory, logging and tracing, secure connection and sequence file selection. For more information, refer to "Managed (J2C) connection factory properties" on page 158 topic.
5. Specify the required security credentials in the **Service Properties** area:
   - To use a J2C authentication alias, select the **Using an existing JAAS alias (recommended)** field, and specify the name of the alias in the **J2C Authentication Data Entry** field. You can specify an existing authentication alias or create one at any time before deploying the module. The name is case sensitive and includes the node name.
   - To use managed connection properties, select the **Using security properties from managed connection factory** field, and type the values in the **User name** and **Password** fields.
   - **User name** - Specifies the name of the user who has privileges to connect to the FTP server and perform FTP operations. For more information, see "User name property (UserName)" on page 219.
   - **Password** - Specifies the password of the user who has privileges to connect to the FTP server and perform FTP operations. For more information, see "Password property (Password)" on page 208

- To administer the user name and password from other mechanism, select **Other**.

6. If you have multiple instances of the adapter, expand **Logging and tracing** and set **Adapter ID** to a value that is unique for this instance. For more information about this property, see “Resource adapter properties” on page 153.
7. Optional: In the Service properties section of the window, specify a Java Authentication and Authorization Services (JAAS) alias for the adapter to use at run time. This is the authentication alias that you set up on the FTP server. The name is case sensitive. For more details, see "Creating the authentication alias".
8. In the **Data format options** field, select one of the following:
   - **Use default data binding 'FTPFileBaseDataBinding' for all operations**

     A non-configured data binding for all the operations used in the service.
   - **Use a data binding configuration for all operations**

     A configured data binding that will be used for all the operations used in the service.
   - **Specify a data binding for each operation**

     No default binding is specified. You can select a specific data binding for each operation used in the service.
9. Optional: Select the **Change the logging properties for the wizard** check box if you want to specify the log file output location or define the level of logging for this module. For information about logging levels, see the section on configuring logging properties in the Troubleshooting and support topic.

Results

The external service wizard now has the information it needs to connect to the FTP server.

What to do next

If you have selected the **Data format options** as either Use default data binding 'FTPFileBaseDataBinding' for all operations or Specify a data binding for each operation, click **Next** to continue to work in the wizard to select a data type for the module and to name the operation associated with the data type.

If you have selected the **Data format options** as Use a data binding configuration for all operations, proceed to Configuring the data binding. “Configuring the data binding and data handler” on page 83.

Selecting a data type and operation name

Use the external service wizard to select a data type and to name the operation associated with the data type. For outbound communications, the external service wizard gives you the choice of three different data types: user-defined type, generic FTP business object, and generic FTP business object with business graph. Each data type corresponds to a business object structure.

Before you begin

Before you can perform the following steps, you must have specified the connection properties for the adapter to connect to the FTP server.

About this task

To select a data type and name the operation associated with it, follow this procedure.

Procedure

1. In the Operations window, click **Add** to create a new operation.
2. In the Operation window, open the **Operation kind** list and select an operation. In this example, the **Create** operation is selected.
3. In the Operation window, select a data type and click **Next**. In this example, the **User defined** data type is selected.

   If you select **User defined type**, you must provide a user-defined data binding to support it. The data bindings provided by the **Generic FTP business object** only support generic input types for the supported operations.
4. Optional: To have the file name returned or to have True or False returned during Delete and ServerToServer operations, select the **Enable response type for the operation** check box. For Exists, List and Retrieve operations, a response type is required, and by default the **Enable response type for the operation** check box is selected.
5. Click **Next**.
6. In the Operation window, type a name for the operation in the **Operation name** field. Name the operation something meaningful. If this module is going to be used to create a new customer record, name it something like `createCustomer`. For more information about the types of operations the adapter can perform, see the Table 1 on page 4 topic.

   Note: Names cannot contain spaces.

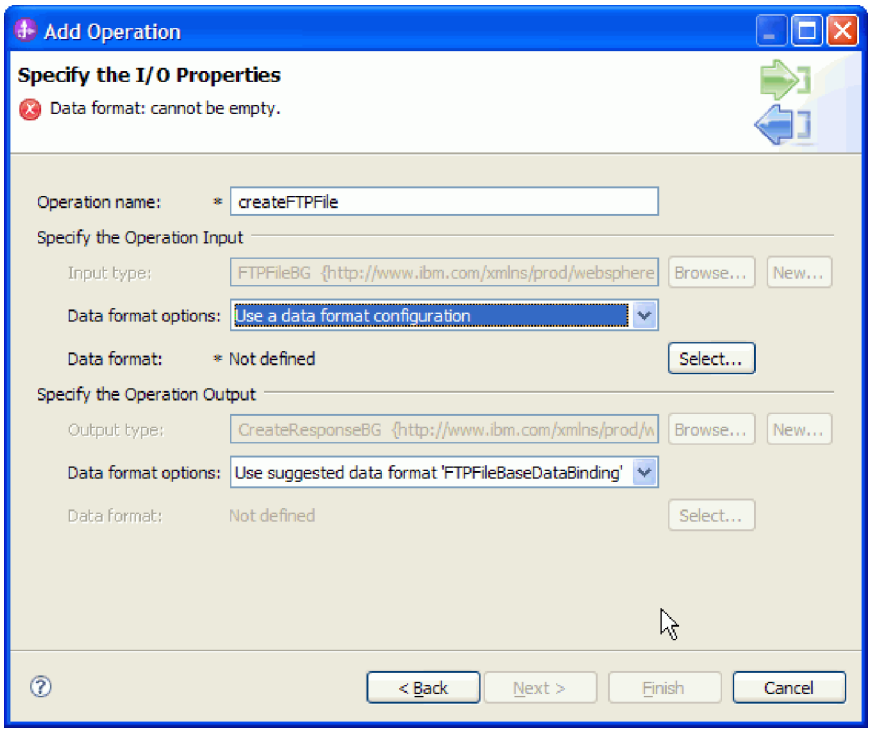


Figure 30. Specify the I/O Properties window

7. In the Specify the Operation Input area, select **New** for the **Input Type** field to create a new data type. To use an existing data type, click **Browse** and select it.

Results

A data type is defined for the module and the operation associated with this data type is named.

What to do next

If you choose to add and configure a data binding to be used with the module, Select **Use a data format configuration from the Data format** options list. Click **Select** next to the Data Format field. Proceed with configuring the data binding with the steps mentioned in Configuring the data binding and data handler topic.

If you choose to use the default data binding, proceed to "Setting interaction specification properties and generating the service" on page 87.

Configuring the data binding and data handler

Each data type has an equivalent data binding that is used to read the fields in a business object and fill the corresponding fields in a file. In the external service wizard, you add a data binding to your module and configure it to correspond with your data type. This way, the adapter knows how to populate the fields in a file with information it receives in the business object.

Before you begin

You must have selected a data type and chosen a configuration name to be associated with the data type.

Note: Data bindings can be configured before running the external service wizard using WebSphere Integration Developer. To do this, select **New → Configure Binding Resource** in WebSphere Integration Developer and complete the data binding windows described in this documentation.

About this task

To add and configure a data binding for the module, follow this procedure.

Procedure

1. In the Select a Data Format Transformation window, select FTPFileBaseDataBinding from the list. To configure a custom data binding, select **Select your custom data format transformation from the workspace** and select the implementation class name. Click **Next**.

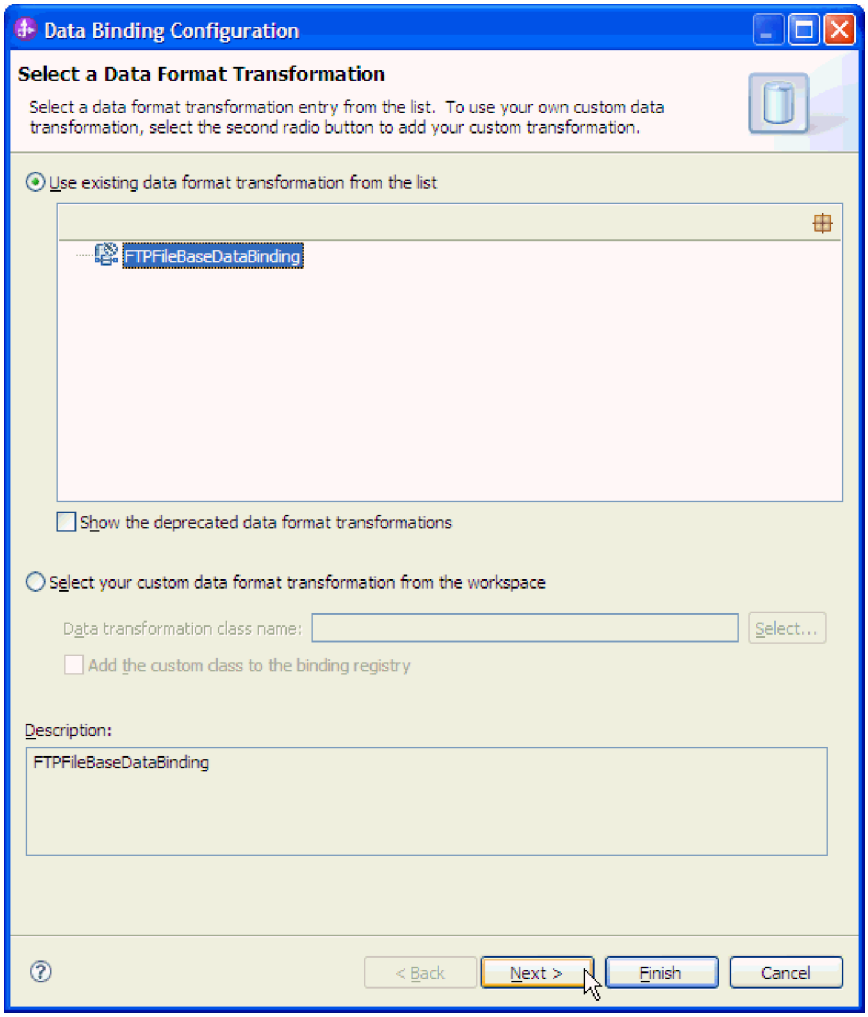


Figure 31. Select a Data Format Transformation window

Specify the data handler which performs the conversions between a business object and a native format when you select a data type that contains the business objects.

2. To configure a data handler, in Specify the Data Transformation Properties window, select the **Binding Type** as DataHandler.
3. Click **Select** next to **Data handler configuration** option.

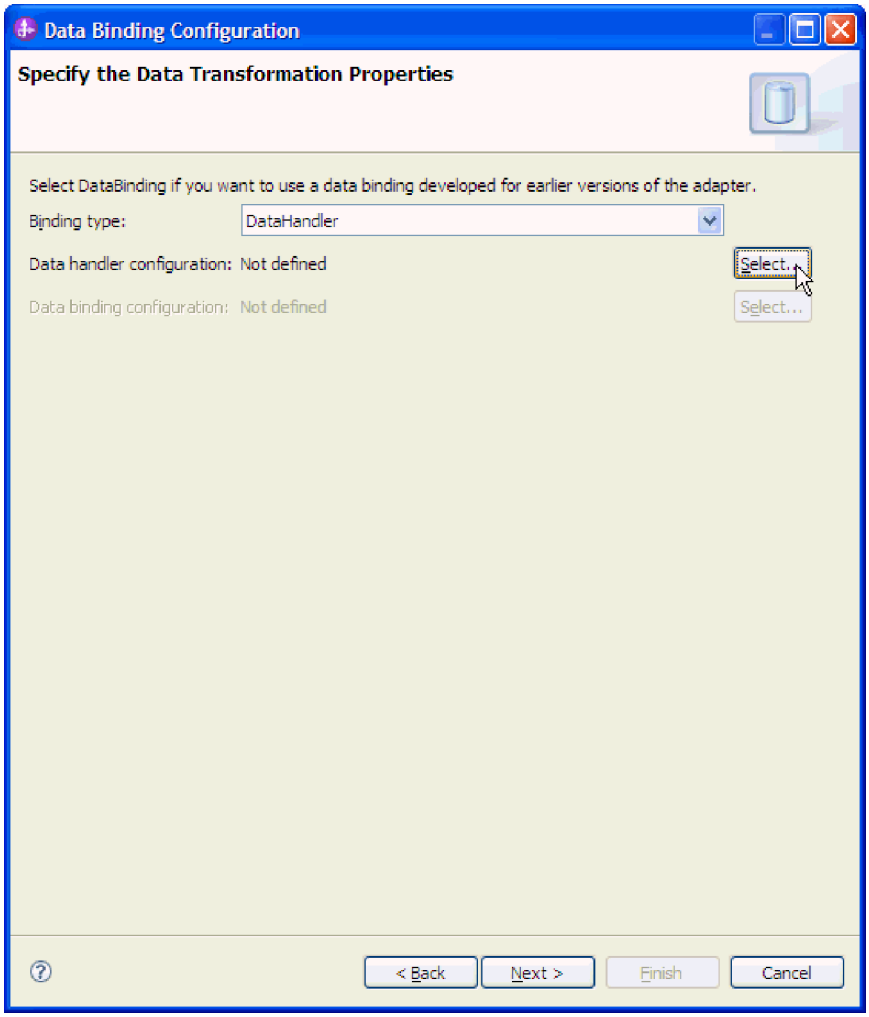


Figure 32. Specify the Data Transformation Properties window

4. In the Select a Data Format Transformation window, select the required Data handler from the list. To configure a custom data handler, select **Select your custom data format transformation from the workspace** and select the implementation class name.

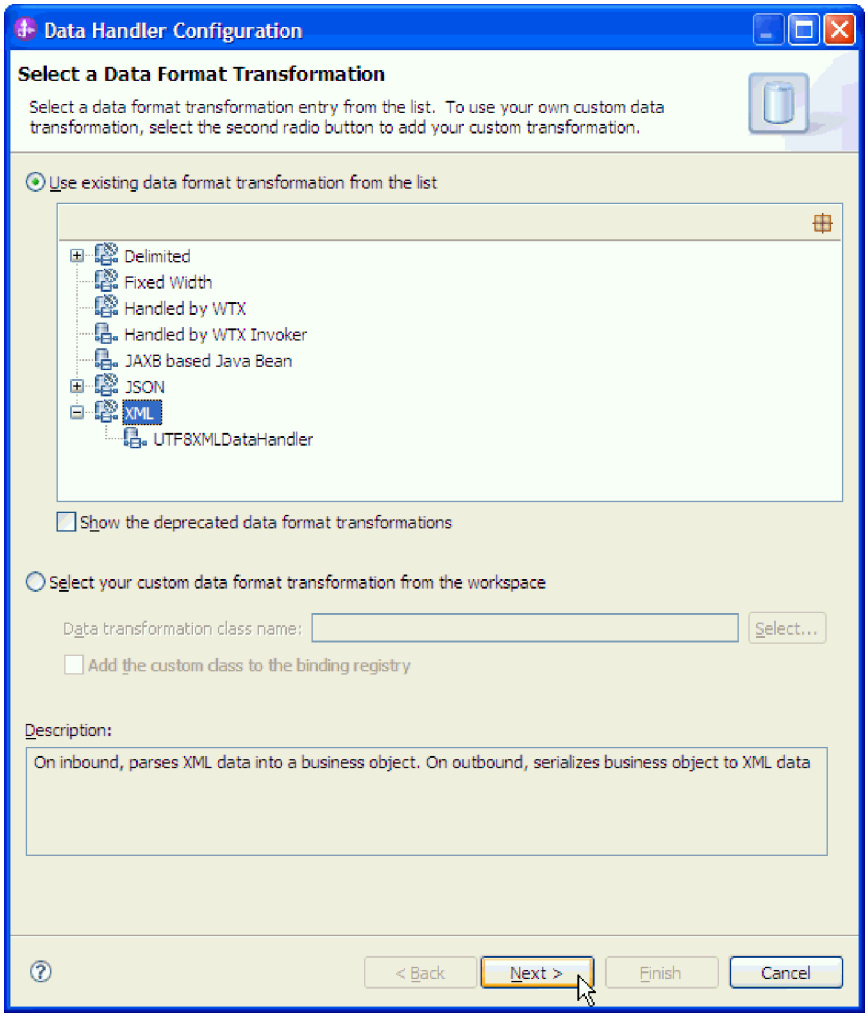


Figure 33. Select a Data Format Transformation window

5. Specify the Module, Namespace, Folder, and Name for the data binding configuration in the Configure a Data Transformation Configuration window.

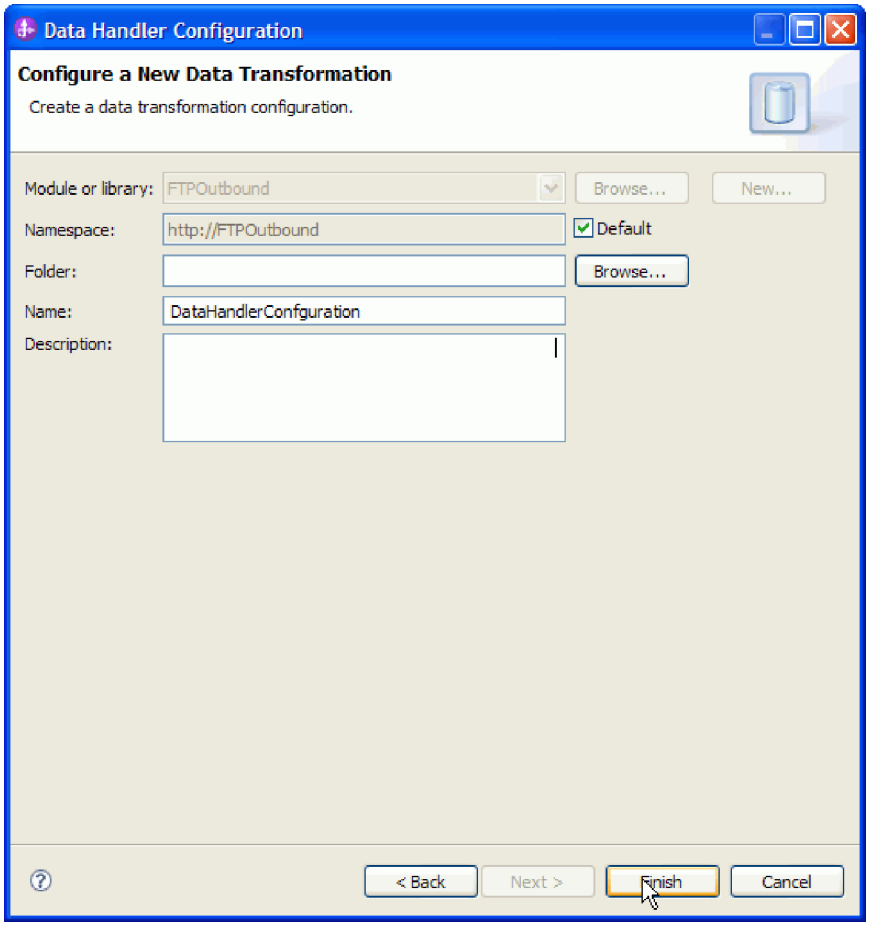


Figure 34. Configure a New Data Transformation window

6. Click **Finish**.

Results

A data binding and data handler is configured for use with the module.

What to do next

From the current external service wizard window, proceed to the next window.

Setting interaction specification properties and generating the service

Interaction specification properties are optional. If you choose to set them, the values you specify appear as defaults in all parent FTP business objects generated by the external service wizard. Interaction specification properties control the interaction for an operation. While creating artifacts for the module, the adapter generates an import file. The import file contains the operation for the top-level business object.

About this task

To set interaction specification properties and generate artifacts, follow this procedure. For more information about interaction specification properties, see the topic about it in this documentation.

Procedure

1. Optional: To set interaction specification properties, populate the fields in the Operations window. You can also click **Advanced** to add additional property details.
   a. Type values for any fields you want to set as defaults.
   b. Click **Next**.

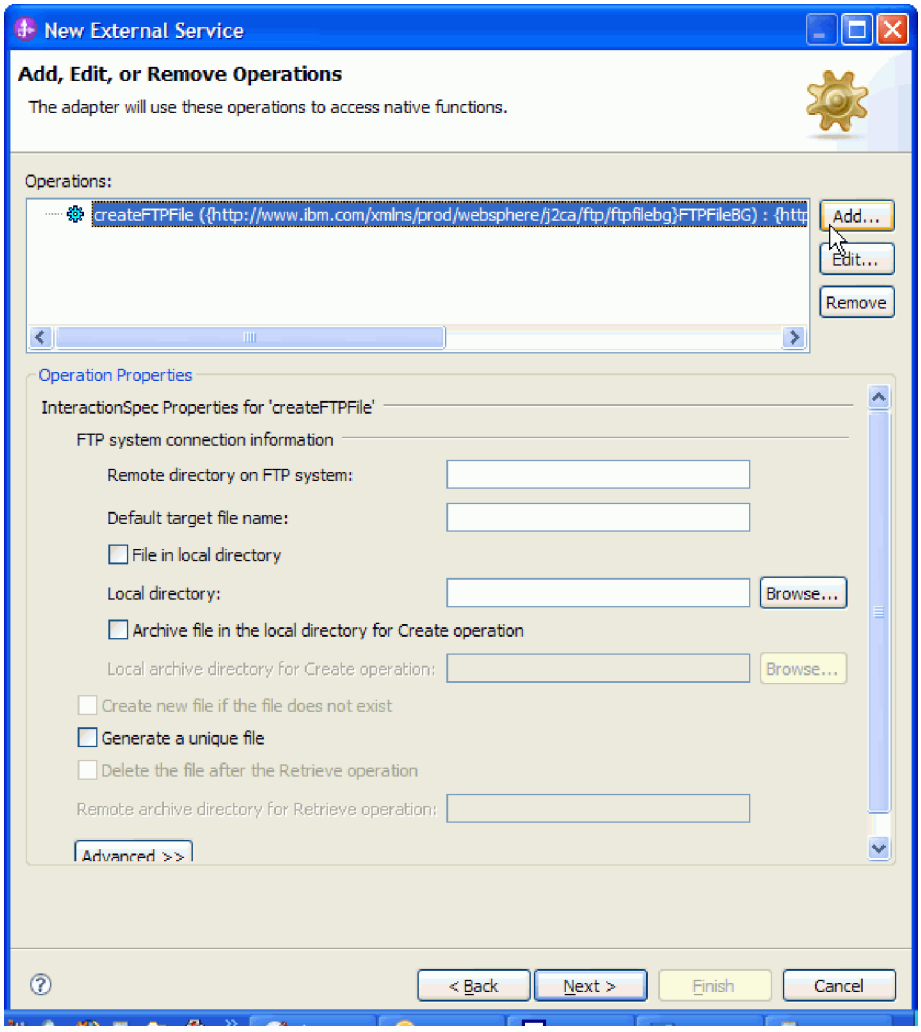


Figure 35. Interaction specification properties

2. On the Generate Service window, supply a name for the interface. This is the name that is displayed in the WebSphere Integration Developer assembly diagram.

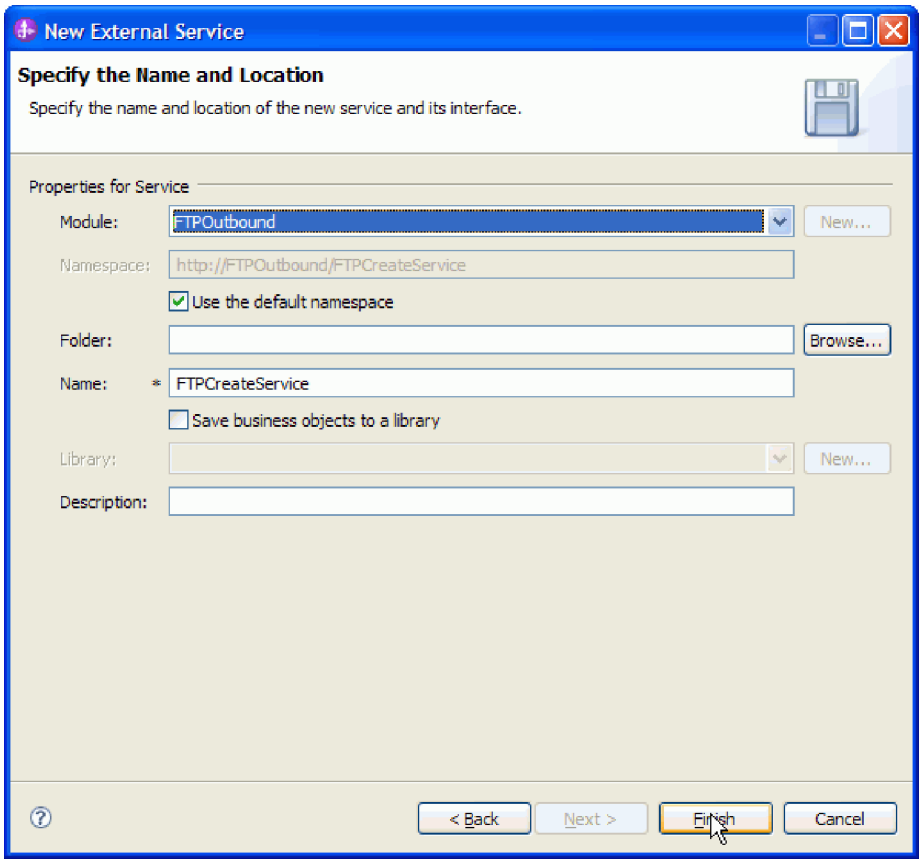


Figure 36. Specify the Name and Location window

3. Click **Finish**. The WebSphere Integration Developer assembly diagram opens and the interface you created is displayed.

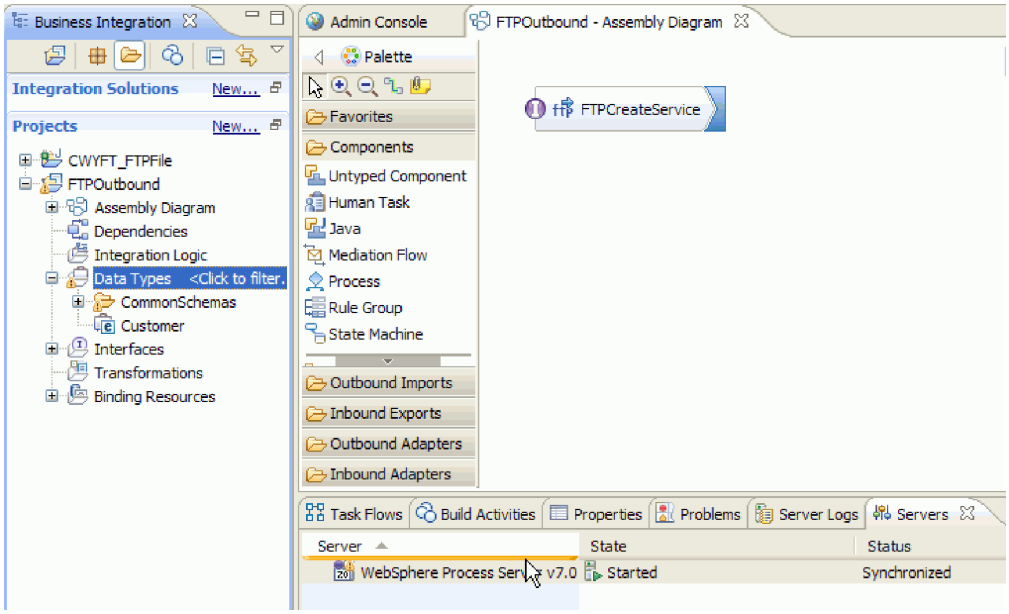


Figure 37. Interface in WebSphere Integration Developer

4. Optional: Repeat the previous steps to add all other required operations, including the bindings, data handlers, and interaction specifications.

Results

WebSphere Integration Developer generates the artifacts and the import. The outbound artifacts that are created are visible in the WebSphere Integration Developer Project Explorer under your module.

What to do next

Deploy the module to the server.

Configuring the module for inbound processing

To configure a module to use the adapter for inbound processing, use the external service wizard in WebSphere Integration Developer to build business services, specify data transformation processing, and generate business object definitions and related artifacts.

Setting deployment and runtime properties

Specify deployment and runtime properties that the external service wizard uses to connect to the FTP server.

Before you begin

Before you can set the properties in this section, you must create your adapter module. It should be displayed in WebSphere Integration Developer below the adapter project. For more information about creating the adapter project, refer to "Starting the external service wizard" on page 77 topic.

About this task

To set deployment and runtime properties, follow this procedure. For more information about the properties in this topic, refer to "Activation specification properties" on page 192 topic.

Procedure

1. In the Processing Direction window, select **Inbound** and click **Next**.
2. In the **Deploy connector project** field, specify whether to include the adapter files in the module. Choose one of the following options:
   - **With module for use by single application**

     With the adapter files embedded in the module, you can deploy the module to any application server. Use an embedded adapter when you have a single module using the adapter or when multiple modules need to run different versions of the adapter. By using an embedded adapter, you can upgrade the adapter in a single module without the risk of destabilizing other modules by changing their adapter version.
   - **On server for use by multiple applications**

     If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central

location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.

3. Define the following FTP system connection information for your module. For more information, refer to "Activation specification properties" on page 192 topic.
   - **Host name** - Specifies the host name of the FTP server.
   - **Remote directory** - Specifies the directory on the FTP server where the adapter polls and picks up files.
   - **Local directory** - Specifies the directory on the adapter workstation where the event files are downloaded from the FTP server.
   - **Protocol** - Specifies the protocol used to connect to the FTP server. Following are the protocols that can be specified:
     - FTP - File Transfer Protocol
     - FTP over SSL - File Transfer Protocol over Secure Socket Layer
     - FTP over TLS - File Transfer Protocol over Transport Layer Security
     - SFTP - Secure shell File Transfer Protocol
   - **Port number** - Specifies the port number of the FTP server.

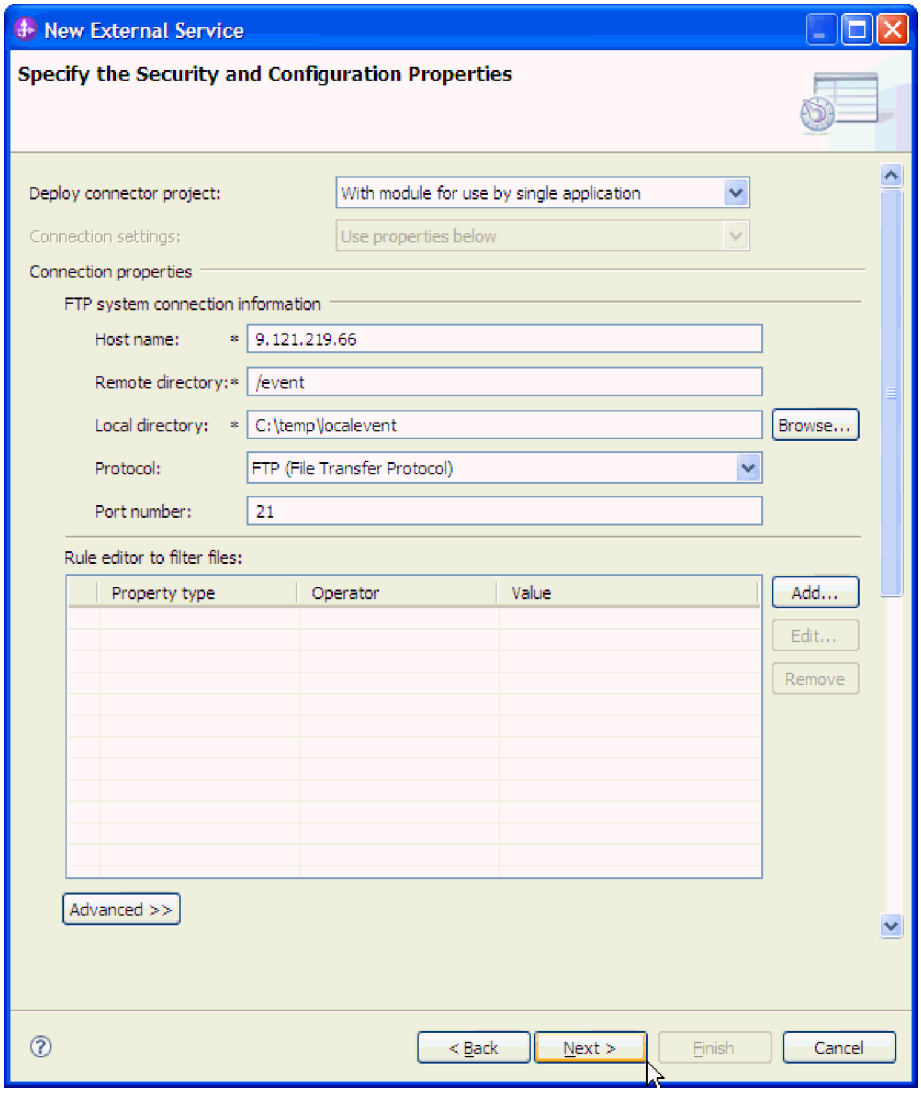


Figure 38. Specify the Security and Configuration Properties window

4. To filter the inbound event file by configuring rules, click **Add** or **Edit** in the Rule editor table. The rule constitutes three parameters, namely, Property type, Operator and Value.

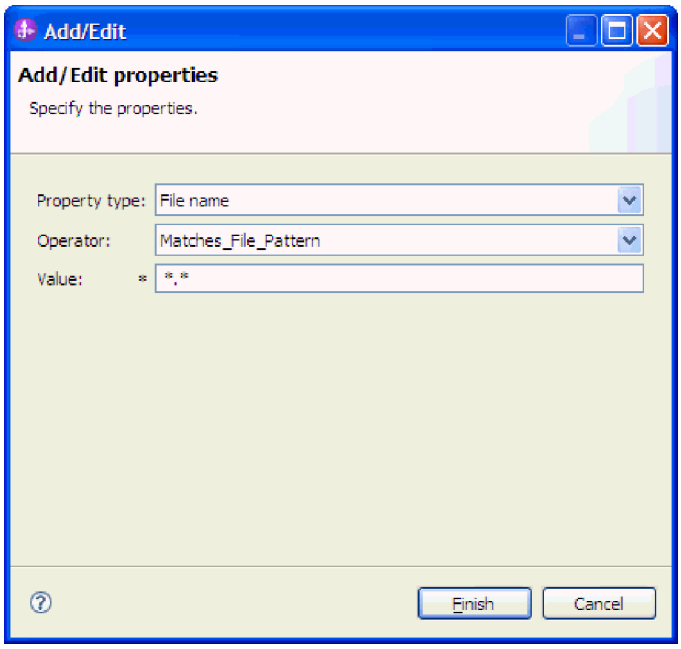


Figure 39. Adding or editing a rule

a. Select any of the following metadata filtering property types from **Property type** list.
   - FileName
   - FileSize
   - LastModified

b. Select the operator for the property type from the **Operator** list. Each of the property type metadata has its own operators.
   1) FileName contains the following operators:
      - Matches_File_Pattern (matches pattern)
      - Matches_RegExp (matches regular expression)
   2) FileSize metadata contains the following operators:
      - Greater than
      - Less than
      - Greater than or equal to
      - Less than or equal to
      - Equal to
      - Not equal to
   3) LastModified metadata contains the following operators:
      - Greater than
      - Less than
      - Greater than or equal to
      - Less than or equal to
      - Equal to
      - Not equal to

c. Type the value for filtering the event file in the **Value** column. You must enter a valid Java regular expression in value for Matches_RegExp operator.

To configure multiple rules, select **END-OF-RULE** option for each rule from the **Property type** list.

Note: The rules are grouped by using the logical **OR** operator, unless **END-OF-RULE** is selected in the property field. If an **END-OF-RULE** is selected between expressions (an expression can be a single rule or multiple rules grouped by an OR operator), it will be grouped using the logical **AND** operator. For example, If the rule A (FileName) is grouped with rule B (FileSize) using the logical **OR** operator, and on selecting the **END-OF-RULE** option, this expression will be grouped with another rule C (LastModified) using an **AND** operator. This can be represented as follows: ((A) OR (B)) AND (C)

For more information see, "Rule editor to filter files" on page 219.

5. Optional: Specify advanced properties by clicking **Advanced**. Expand each of the advanced sections to review the properties.
   - Event polling configuration
   - Event delivery configuration
   - Event persistence configuration
   - Additional configuration
   - FTP archiving configuration
   - Socks proxy server connection information
   - Secure configuration
   - Bidi properties
   - Logging and tracing properties

   The following sections describe the options that are available in the advanced property groups.
   - **Event polling configuration**
     a. In the **Interval between polling periods** field, specify the number of milliseconds that the adapter should wait between polling periods. For more information, see "Interval between polling periods (PollPeriod)" on page 209.
     b. In the **Maximum events in polling period** field, specify the number of events that the adapter should deliver in each polling period. For more information, see "Maximum events in polling period (PollQuantity)" on page 209.
     c. In the **Retry interval if connection fails** field, specify the number of milliseconds for the adapter to wait before trying to connect after a connection failure during polling. For more information, see "Retry interval if connection fails (RetryInterval)" on page 214.
     d. In the **Number of times to retry the system connection** field, specify the number of times to retry the connection before reporting a polling error. For more information, see "Number of times to retry the system connection (RetryLimit)" on page 214.
     e. If you want the adapter to stop if polling errors occur, select **Stop the adapter when an error is encountered while polling**. If you do not select this option, the adapter logs an exception but continues to run. For more information, see "Stop the adapter when an error is encountered while polling (StopPollingOnError)" on page 218.
     f. Select **Retry EIS connection on startup** if you want the adapter to retry a failed connection when starting. For more information, see "Retry EIS connection on startup (RetryConnectionOnStartup)" on page 213.

- **Event delivery configuration**
  a. In the **Type of delivery** field, select the delivery method. The methods are described in "Delivery type (DeliveryType)" on page 198.
  b. If you want to ensure that events are delivered only once and to only one export, select **Ensure once-only delivery**. This option might reduce performance but does not result in duplicate or missing an event delivery. For more information, see "Ensure once-only event delivery (AssuredOnceDelivery)" on page 196.
  c. In the **Retry limit for failed events** field, specify the number of times that the adapter will attempt to redeliver an event before marking it as failed. For more information, see "Retry limit for failed events (FailedEventRetryLimit)" on page 205.
- **Event persistence configuration**
  a. Select **Auto create event table** if you want the adapter to create the Event Persistence table. For more information, see "Auto create event table property (EP_CreateTable)" on page 197.
  b. In the **Event recovery table name** field, specify the name of the table that the adapter uses for event persistence. For more information, see "Event recovery table name property (EP_EventTableName)" on page 200
  c. In the **Event recovery data source (JNDI) name** field, specify the JNDI name of the data source that event persistence will use to connect to the JDBC database. For more information, see "Event recovery data source (JNDI) name property (EP_DataSource_JNDIName)" on page 199
  d. In the **User name used to connect to event data source** field, specify the user name that the event persistence will use to connect to the database from the data source. For more information, see "User name used to connect to event data source property (EP_UserName)" on page 219
  e. In the **Password used to connect to event data source** field, specify the password that the event persistence will use to connect to the database from the data source. For more information, see "Password used to connect to event data source property (EP_Password)" on page 208.
  f. In the **Database schema name** field, specify the schema name of the database that the event persistence will use. For more information, see "Database schema name property (EP_SchemaName)" on page 198.
- **Additional configuration**
  a. In the **Retrieve files with this pattern** field, specify the filter for the event files. For more information, see "Retrieve files with this pattern property (EventFileMask)" on page 212.
  b. In the **Sort event files** field, specify the sorting order of the event files being polled. For more information, see "Sort event files property (SortEventFiles)" on page 216.
  c. Select the **Enable remote verification** check box to enable remote verification. This property checks if the control and data connections are established with the same host (typically, the machine from which you establish a connection to the FTP server). The connection fails if the control and data connections are not established. By default, the **Enable remote verification** check box is selected.

     Note: This property is applicable only to FTP and FTPS protocols. For more information, see "Enable remote verification property (enableRemoteVerification)" on page 212

d. In the **Encoding used by FTP server** field, specify the encoding of the FTP server. For more information, see “Encoding used by FTP server property (EISEncoding)” on page 199.

e. In the **File content encoding** field, specify the encoding used to read the event files. For more information, see “File content encoding property (FileContentEncoding)” on page 201.

f. In the **FTP server connection mode** field, specify the data connection mode used by the FTP server during file transfers. For more information, see “FTP server connection mode property (DataConnectionMode)” on page 200.

g. In the **File transfer type** field, specify the file transfer type used during inbound processing. For more information, see “File transfer type property (FileTransferType)” on page 204.

h. In the **Number of files to get at a time** field, specify the number of files retrieved from the remote FTP URL. For more information, see “Number of files to get at a time property (ftpGetQuantity)” on page 204.

i. In the **Number of poll periods between downloads** field, specify how frequently the adapter polls the FTP server. For more information, see “Number of poll periods between downloads property (ftpPollFrequency)” on page 204.

j. In the **Custom parser class name** field, specify the fully qualified class name of the custom parser that is used to parse the ls output. For more information, see “Custom parser class name property (CustomParserClassName)” on page 197.

k. Select **Pass only file name and directory, not the content** to specify that the file content of the event file is not sent to the export. For more information, see “Pass only file name and directory, not the content property (FilePassByReference)” on page 204.

l. Select **Include business object delimiter in the file content** to specify that the delimiter will be sent with the business object content for further processing. For more information, see “Include business object delimiter in the file content property (IncludeEndBODelimiter)” on page 206

m. Select **Split file content based on the size (bytes) or delimiter** to use the size in bytes or the delimiter to split the file content. For more information, see “Splitting function class name property” on page 218

n. In the **Specify criteria to split file content** field, specify that different values will be taken, based on the value of the SplittingFunctionClassName property. For more information, see “Specify criteria to split file content property (SplitCriteria)” on page 216.

o. In the **Split function class name** field, specify the fully qualified class name of the class file to be used to enable file splitting. For more information, see “Splitting function class name property” on page 218.

p. In the **Run FTP script file before downloading files** field, specifies the path of the script file that will be executed before downloading the files from the FTP server. For more information, see “Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)” on page 206.

q. In the **Run FTP script file after downloading files** field, specifies the path of the script file that will be executed after downloading the files from the FTP server. For more information, see “Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)” on page 205.

- **FTP archiving configuration**
  a. In the **Local archive directory** field, specify the absolute path of the local Archive directory. For more information, see "Local archive directory property (LocalArchiveDirectory)" on page 206.
  b. In the **File extension for local archive** field, specify the file extension used to archive the original event file. For more information, see "File extension for local archive property (originalArchiveExt)" on page 208.
  c. In the **Success file extension for local archive** field, specify the file extension used to archive all the successfully processed business objects. For more information, see "Success file extension for local archive property (SuccessArchiveExt)" on page 219.
  d. In the **Failure file extension for local archive** field, specify the file extension used to archive business objects in the event file that are not successfully processed. For more information, see "Failure file extension for local archive property (FailedArchiveExt)" on page 201.
  e. In the **Remote archive directory** field, specify the directory. For more information, see "Remote archive directory property (ftpArchiveDirectory)" on page 211.
  f. In the **File extension for remote archive** field, specify the file extension or suffix that the adapter uses to rename the remote FTP file. For more information, see "File extension for remote archive property (ftpRenameExt)" on page 201.
- **Socks proxy server connection information**
  a. In the **Host name** field, specify the host name of the machine used as a proxy server through which the adapter requests are routed to the FTP server. For more information, see "Host name property (SocksProxyHost)" on page 215.
  b. In the **Port number** field, specify the port number of the proxy server through which the adapter requests are routed to the FTP server. For more information, see "Port number property (SocksProxyPort)" on page 216.
  c. In the **User name** field, specify the user name for authenticating the proxy server. For more information, see "User name property (SocksProxyUserName)" on page 216.
  d. In the **Password** field, specify the password used to authenticate the proxy server. For more information, see "Password property (SocksProxyPassword)" on page 215.
- **Secure configuration**
  a. If you want to compare the host key of the SFTP server with the host keys known to the adapter:
     1) Select the **Enable remote server authentication for SFTP protocol** check box. The host key file has to be available with the host keys of the trusted server before the first attempt to connect to SFTP server is made. For more information, see Enable server verification property (EnableServerVerification)
     2) In the **Host key file** field, specify the absolute file path to the host key file. The host key file is created by the administrator and contains the host keys of all the trusted servers. The Host key file property points to the file on the adapter workstation. For more information, see Host key file property (HostKeyFile)
  b. If you want to enable public key authentication, specify the following properties:

1) In the **Private key file** field, specify the private key used to authenticate to the Secure shell server. For more information, see “Private key file property (PrivateKeyFilePath)” on page 210.
2) In the **Passphrase** field, specify the phrase used for enhanced security by encrypting the private key. For more information, see Passphrase property (Passphrase)

c. Specify the following properties for the FTPS protocol:
1) In the **FTPS connection mode** field, specify the connection mode (Implicit or Explicit) to connect to the FTPS server, when FTPS is selected as protocol. For more information, see FTPS connection mode property (ftpsConnectionMode).
2) In the **Data channel protection level** field, select the level of the data channel protection that you want to use:
   – Select **Private**, if the data transfer between the Adapter and the FTPS server has to be in an encrypted form.
   – Select **Clear**, if the data transfer between the Adapter and the FTPS server has to be in clear text form.

   For more information, see “Data channel protection level (dataProtectionLevel)” on page 162.
3) In the **Keystore type** field, specify type of the keystore. For more information, see Keystore type property (keyStoreType).
4) In the **Truststore file** field, specify the path of the truststore file that contains the certificates of the servers trusted by the adapter. For more information, see Truststore file property (trustStorePath).
5) In the **Truststore password** field, specify the password of the truststore file. It is used to check the integrity of the truststore data. If this value is not specified, the integrity check will not be performed. For more information, see Truststore password property (trustStorePassword).
6) In the **Keystore file** field, specify the path of the keystore file. The Keystore file contains the private key entry of the FTPS client and also contains a certificate chain for the corresponding public key. For more information, see Keystore file property (keyStorePath).

   Note: Both Keystore file and Truststore file properties share the properties of Keystore type.
7) In the **Keystore password** field, specify the password of the keystore. It is used to check the integrity of the keystore data. If this value is not specified, integrity check will not be performed. For more information, see Keystore password property (keyStorePassword).
8) In the **Key password** field, specify the password of the key that is used to recover the keys from the keystore. For more information, see Key password property (keyPassword).

- **Bidi properties**
- **Logging and tracing**

a. If you have multiple instances of the adapter, expand and set Adapter ID to a value that is unique for this instance. For more information about this property, see “Resource adapter properties” on page 153.
b. Select **Disguise user data as 'XXX' in log and trace files** if you want to prevent sensitive user data from being written to log and trace files. For more information, see “Disguise user data as "XXX" in log and trace files (HideConfidentialTrace) ” on page 156.

6. Specify the required security credentials in the **Service Properties** area:
   - To use a J2C authentication alias, select the **Using an existing JAAS alias (recommended)** field, and specify the name of the alias in the **J2C Authentication Data Entry** field. You can specify an existing authentication alias or create one at any time before deploying the module. The name is case sensitive and includes the node name.
   - To use activation specification properties, select the **Using security properties from the activation specification** field, and type the values in the **User name** and **Password** fields.
   - **User name** - Specifies the name of the user who has privileges to connect to the FTP server and perform FTP operations. For more information, see "User name property (UserName)" on page 219.
   - **Password** - Specifies the password of the user who has privileges to connect to the FTP server and perform FTP operations. For more information, see "Password property (Password)" on page 208.
   - To administer the user name and password from other mechanism, select **Other**.
7. Select one of the options from the **Function selector** field. A function selector assigns incoming messages or requests to the correct operation on the service.
   - **Function selector options**

     For example, select **Use a Function Selector configuration**. If choosing to use this option, click **Next**.
   - **Function selector**

     If choosing this option, complete the following steps:

     a. Click **Select** next to the **Function Selector** field.

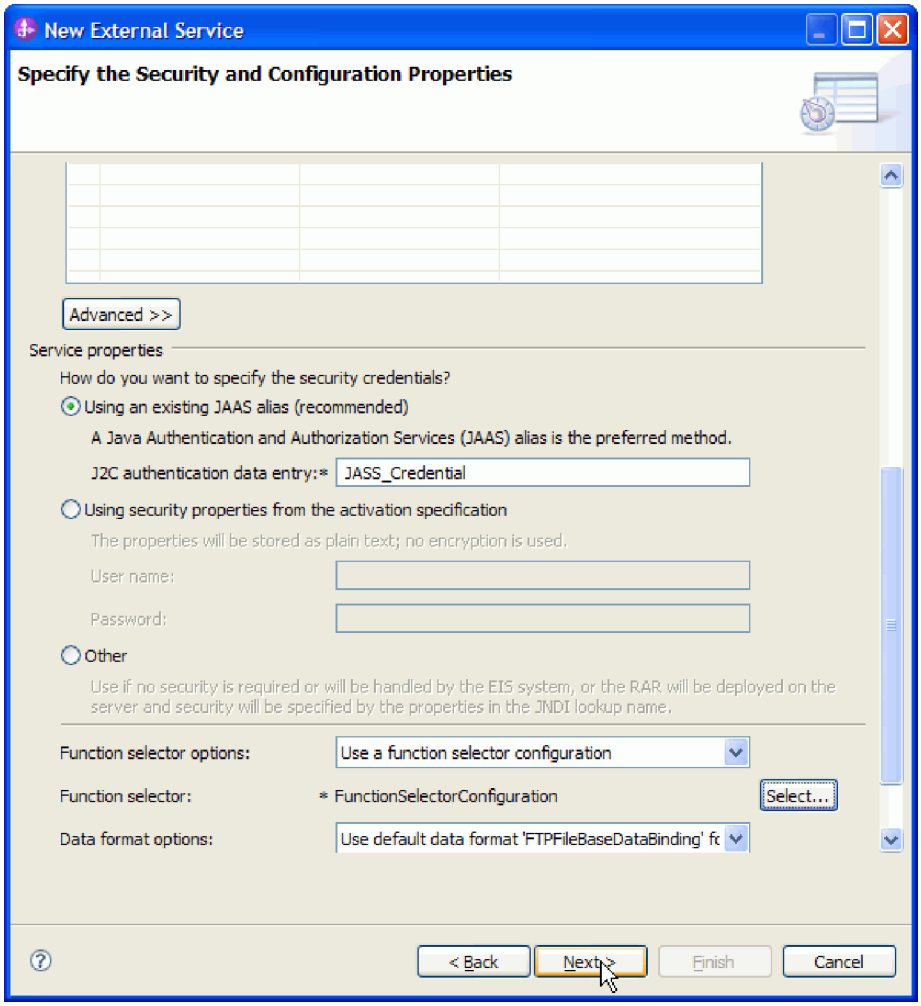


Figure 40. Specify the Security and Configuration Properties window

b. In the Select Function Selector window, select the option, **Use existing function selector from the list**. A list of available function selectors is displayed. Select the function selector (this example uses FilenameFunctionSelector properties). Click **Next**.

   Note: The EIS function name is not available in the external service wizard. If you want to specify a value other than the default that is generated by the adapter, you can edit it using the assembly editor.

8. Click **Finish** in the New Function Selector Configuration window.
9. Click **Next** in the Service Configuration Properties window.

Results

The external service wizard now has the information it needs to connect to the FTP server.

What to do next

If you have selected the **Data format options** as either Use default data binding 'FTPFileBaseDataBinding' for all operations or Specify a data binding for each

operation, click **Next** to continue to work in the wizard to select a data type for the module and to name the operation associated with the data type.

If you have selected the **Data format options** as Use a data binding configuration for all operations, proceed to Configuring the data binding. “Configuring the data binding and data handler” on page 83.

Selecting a data type and operation name

Use the external service wizard to select a data type and to name the operation associated with the data type. For inbound communications, the external service wizard gives you the choice of three different data types: user-defined type, generic FTP business object, and generic FTP business object with business graph. Each data type corresponds to a business object structure.

Before you begin

You must have specified the connection properties for the adapter to connect to the FTP server before you can complete the following steps.

About this task

To select a data type and name the operation associated with it, follow this procedure.

Procedure

1. In the Operations window, click **Add**.
2. In the Add Operations window, select **The data type for the operation input**, and click **Next**. If you select **User defined type**, you must provide a user-defined data binding to support it. The **Generic FTP business object** provided data binding only supports generic input types for the supported operations.
3. In the Operation window, type a name in the **Operation name** field or keep the default `emitFTPFile` name.

   Note: Names cannot contain spaces.

Results

A data type is defined for the module and the operation associated with the data type is named.

What to do next

If you choose to add and configure a data binding to be used with the module, Select **Use a data format configuration from the Data format** options list. Click **Select** next to the Data Format field. Proceed with configuring the data binding with the steps mentioned in Configuring the data binding and data handler topic.

If you choose to use the default data binding, proceed to “Generating the service” on page 106.

Configuring the data binding and data handler

Each data type has an equivalent data binding that is used to read the fields in a business object and fill the corresponding fields in a file. In the external service

wizard, you add a data binding to your module and configure it to correspond with your data type. This way, the adapter knows how to populate the fields in a file with information it receives in the business object.

Before you begin

You must have selected a data type and chosen a configuration name to be associated with the data type.

Note: Data bindings can be configured prior to running the external service wizard using WebSphere Integration Developer. To do this, select **New → Configure Binding Resource** in WebSphere Integration Developer and complete the data binding windows described in this documentation.

About this task

To add and configure a data binding for the module, follow this procedure.

Procedure

1. In the Select a Data Format Transformation window, select FTPFileBaseDataBinding from the list. To configure a custom data binding, select **Select your custom data format transformation from the workspace** and select the implementation class name. Click **Next**.

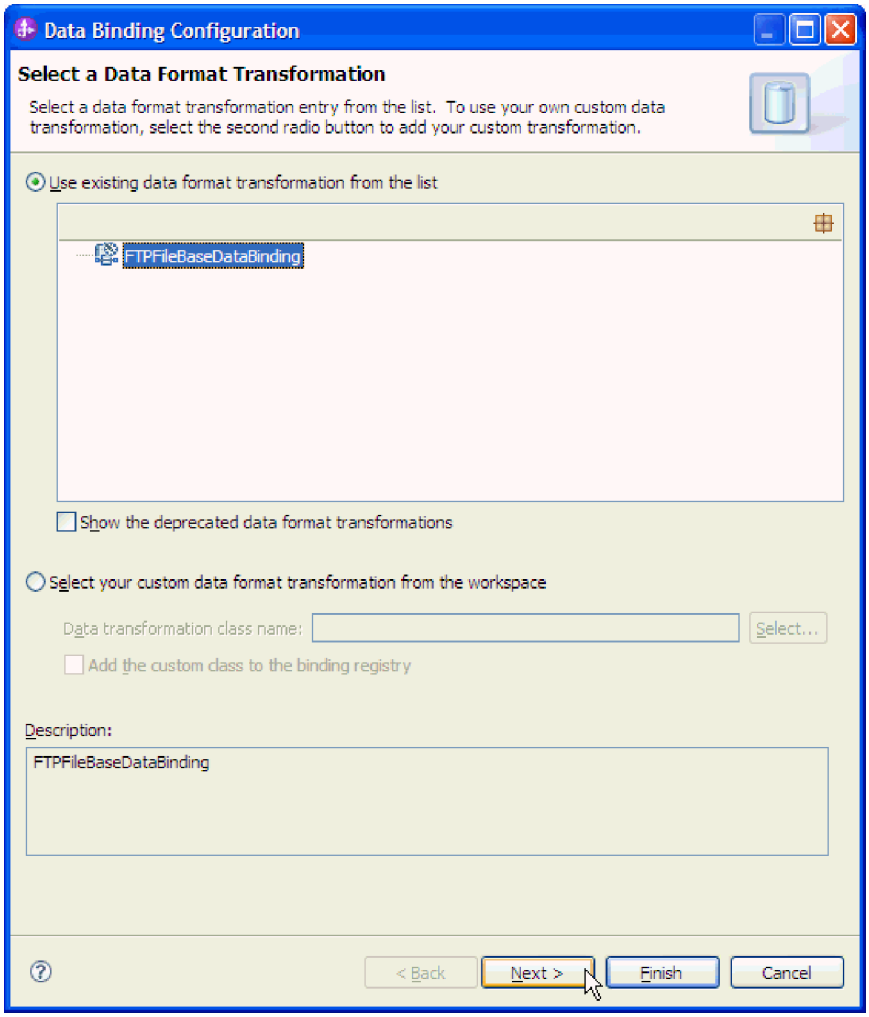


Figure 41. Select a Data Format Transformation window

Specify the data handler which performs the conversions between a business object and a native format when you select a data type that contains the business objects.

2. To configure a data handler, in Specify the Data Transformation Properties window, select the **Binding Type** as DataHandler.
3. Click **Select** next to **Data handler configuration** option.

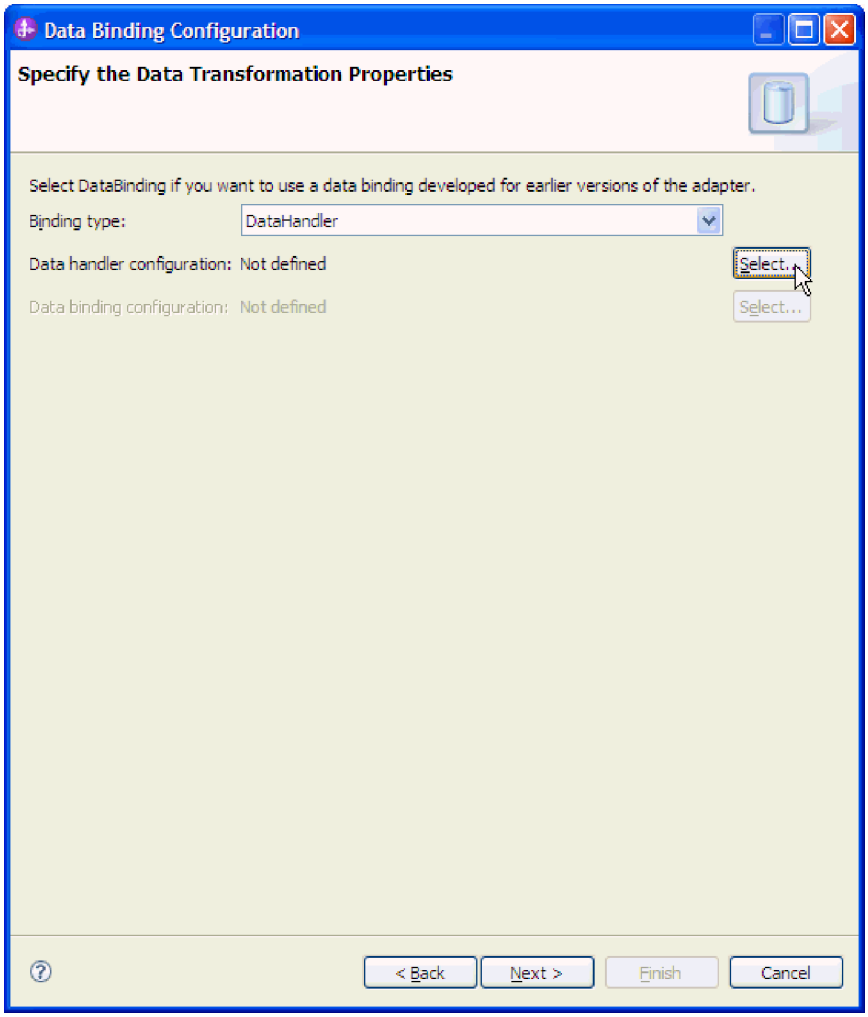


Figure 42. Specify the Data Transformation Properties window

4. In the Select a Data Format Transformation window, select the required Data handler from the list. To configure a custom data handler, select **Select your custom data format transformation from the workspace** and select the implementation class name.

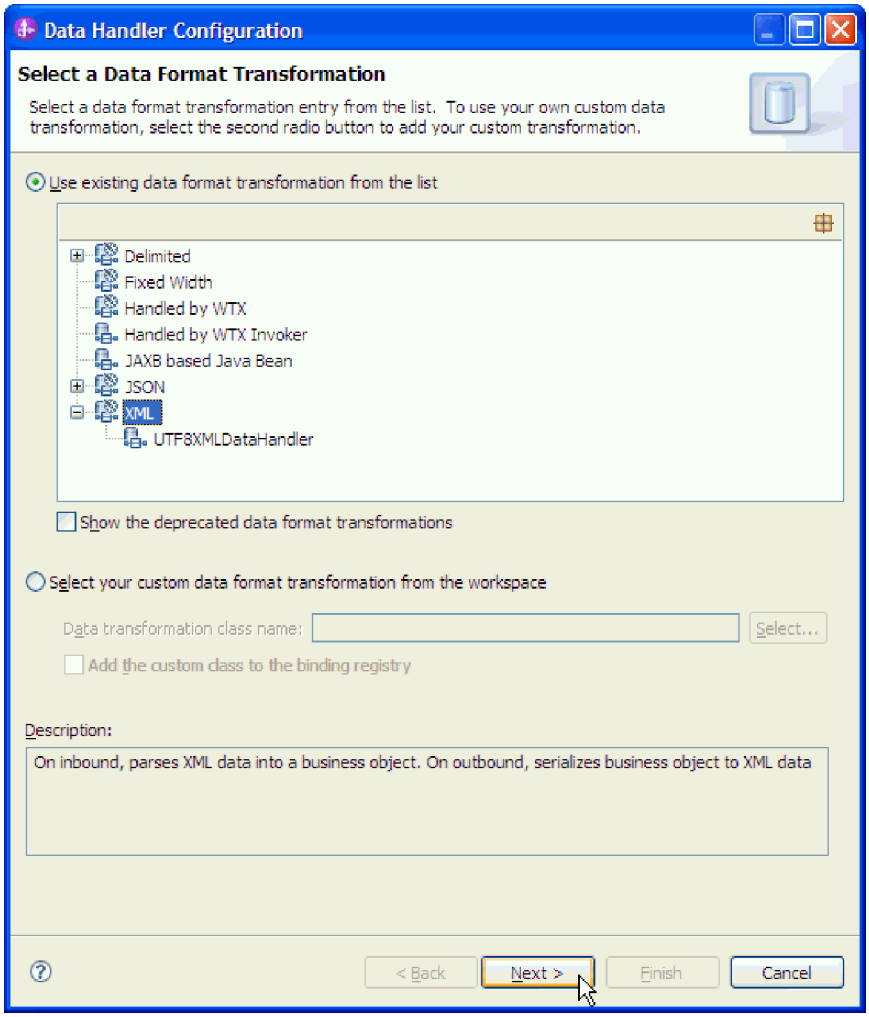


Figure 43. Select a Data Format Transformation window

5. Specify the Module, Namespace, Folder, and Name for the data binding configuration in the Configure a New Data Transformation window.

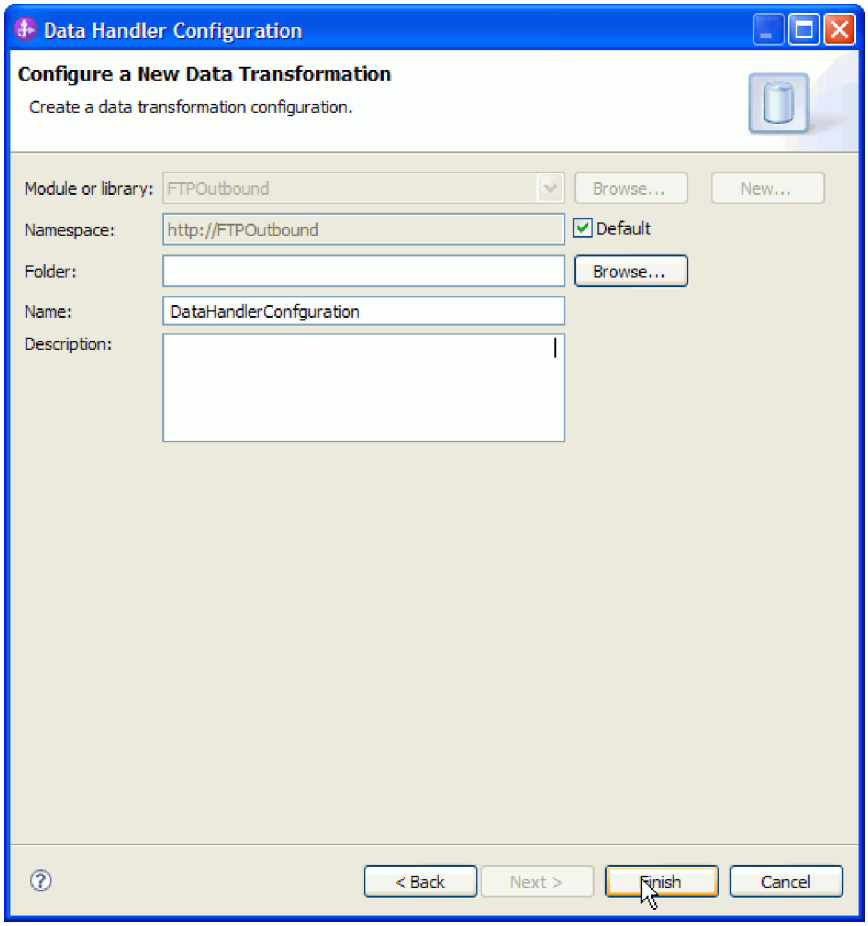


Figure 44. Configure a New Data Transformation window

6. Click **Finish**.

Results

A data binding and data handler is configured for use with the module.

What to do next

From the current external service wizard window, proceed to the next window.

Generating the service

While creating artifacts for the module, the adapter generates an export file. The export file contains the operation for the top-level business object.

About this task

To generate artifacts, follow this procedure.

Procedure

1. Click **Next** in the Operations window.
2. In the Generate Service window, supply a name for the interface. This is the name that will display in the WebSphere Integration Developer assembly diagram.

3. Click **Finish**. The WebSphere Integration Developer assembly diagram opens and the interface you created is displayed.

Results

The WebSphere Integration Developer generates the artifacts and an export. The inbound artifacts that are created are visible in the WebSphere Integration Developer Project Explorer under your module.

What to do next

Deploy the module to the server.

Chapter 5. Changing interaction specification properties using the assembly editor

To change interaction specification properties for your adapter module after generating the service, use the assembly editor in WebSphere Integration Developer.

Before you begin

You must have used the external service wizard to generate a service for the adapter.

About this task

You might want to change interaction specification properties after you have generated a service for the adapter. Interaction specification properties, which are optional, are set at the method level, for a specific operation on a specific business object. The values you specify appear as defaults in all parent business objects generated by the external service wizard. You can change these properties before you export the EAR file. You cannot change these properties after you deploy the application.

To change the interaction specification properties, use the following procedure.

Procedure

1. From the Business Integration perspective of WebSphere Integration Developer, expand the module name.
2. Expand **Assembly Diagram** and double-click the interface.
3. Click the interface in the assembly editor. (It shows the module properties if you do not do the extra click.)
4. Click the **Properties** tab. (You can also right-click the interface in the diagram and click **Show in Properties**)
5. Under **Binding**, click **Method bindings**. The methods for the interface are displayed, one for each combination of business object and operation.
6. Select the method whose interaction specification property you want to change.
7. Click **Advanced** and change the property in the **Generic** tab. Repeat this step for each method whose interaction specification property you want to change.

Results

The interaction specification properties associated with your adapter module are changed.

What to do next

Deploy the module.

Chapter 6. Deploying the module

Deploy a module to place the files that make up your module and adapter into an operational environment for production or testing. In WebSphere Integration Developer, the integrated test environment features runtime support for WebSphere Process Server or WebSphere Enterprise Service Bus, or both, depending on the test environment profiles that you selected during installation.

Deployment environments

There are test and production environments into which you can deploy modules and adapters.

In WebSphere Integration Developer, you can deploy your modules to one or more servers in the test environment. This is typically the most common practice for running and testing business integration modules. However, you can also export modules for server deployment on WebSphere Process Server or WebSphere Enterprise Service Bus as EAR files using the administrative console or command-line tools.

Deploying the module for testing

In WebSphere Integration Developer, you can deploy a module that includes an embedded adapter to the test environment and work with server tools that enable you to perform such tasks as editing server configurations, starting, and stopping servers and testing the module code for errors. The testing is generally performed on the interface operations of your components, which enables you to determine whether the components are correctly implemented and the references are correctly wired.

Generating and wiring a target component for testing inbound processing

Before deploying to the test environment a module that includes an adapter for inbound processing, you must first generate and wire a target component. This target component serves as the *destination* to which the adapter sends events.

Before you begin

You must have generated an export module, using the external service wizard.

About this task

Generating and wiring a target component for inbound processing is required in a testing environment only. It is not necessary when deploying the adapter in a production environment.

The target component receives events. You *wire* the export to the target component (connecting the two components) using the assembly editor in WebSphere Integration Developer. The adapter uses the wire to pass event data (from the export to the target component).

Procedure

1. Create the target component
   a. From the Business Integration perspective of WebSphere Integration Developer, expand **Assembly Diagram** and double-click the export component. If you did not change the default value, the name of the export component is the name of your adapter + **InboundInterface**.

      An interface specifies the operations that can be called and the data that is passed, such as input arguments, returned values, and exceptions. The **InboundInterface** contains the operations required by the adapter to support inbound processing and is created when you run the external service wizard.
   b. Create a new component by expanding **Components**, selecting **Untyped Component**, and dragging the component to the Assembly Diagram.

      The cursor changes to the placement icon.
   c. Click the component to have it displayed in the Assembly Diagram.
2. Wire the components.
   a. Click and drag the export component to the new component.
   b. Save the assembly diagram. Click **File** → **Save**.
3. Generate an implementation for the new component.
   a. Right-click on the new component and select **Generate Implementation** → **Java**.
   b. Select **(default package)** and click **OK**. This creates an endpoint for the inbound module.

      The Java implementation is displayed in a separate tab.
   c. **Optional:** Add print statements to print the data object received at the endpoint for each of the endpoint methods.
   d. Click **File** → **Save** to save the changes.

What to do next

Continue deploying the module for testing.

Adding the module to the server

In WebSphere Integration Developer, you can add modules to one or more servers in the test environment.

Before you begin

If the module you are testing uses an adapter to perform inbound processing, generate and wire a *target component* to which the adapter sends the events.

About this task

In order to test your module and its use of the adapter, you need to add the module to the server.

Procedure

1. *Conditional:* If there are no servers in the **Servers** view, add and define a new server by performing the following steps:
   a. Place your cursor in the **Servers** view, right-click and select **New** → **Server**.
   b. From the Define a New Server window, select the server type.

c. Configure server's settings.
d. Click **Finish** to publish the server.

2. Add the module to the server.
   a. Switch to the servers view. In WebSphere Integration Developer, select **Windows → Show View → Servers**.
   a. Start the server. In the **Servers** tab in the lower-right pane of the WebSphere Integration Developer screen, right-click on the server, and then select **Start**.
3. When the server status is *Started*, right-click on the server, and select **Add and Remove Projects**.
4. In the Add and Remove Projects screen, select your project and click **Add**. The project moves from the **Available projects** list to the **Configured projects** list.
5. Click **Finish**. This deploys the module on the server.

   The Console tab in the lower-right pane displays a log while the module is being added to the server.

What to do next

Test the functionality of your module and the adapter.

Testing the module for outbound processing using the test client

Test the assembled module and adapter for outbound processing using the WebSphere Integration Developer integration test client.

Before you begin

You need to add the module to the server first.

About this task

Testing a module is performed on the interface operations of your components, which enables you to determine whether the components are correctly implemented and the references are correctly wired.

Procedure

1. Select the module you want to test, right-click on it, and select **Test → Test Module**.
2. For information about testing a module using the test client, see the *Testing modules and components* topic in the WebSphere Integration Developer information center.

What to do next

If you are satisfied with the results of testing your module and adapter, you can deploy the module and adapter to the production environment.

Deploying the module for production

Deploying a module created with the external service wizard to WebSphere Process Server or WebSphere Enterprise Service Bus in a production environment is a two-step process. First, you export the module in WebSphere Integration Developer as an enterprise archive (EAR) file. Second, you deploy the EAR file using the WebSphere Process Server or WebSphere Enterprise Service Bus administrative console.

Installing the RAR file (for modules using stand-alone adapters only)

If you chose not to embed the adapter with your module, but instead choose to make the adapter available to all deployed applications in the server instance, you need to install the adapter in the form of a RAR file to the application server. A RAR file is a Java archive (JAR) file that is used to package a resource adapter for the Java 2 Connector (J2C) architecture.

Before you begin

You must set **Deploy connector project** to **On server for use by multiple adapters** in the Specify the Service Generation and Deployment Properties window of the external service wizard.

About this task

Installing the adapter in the form of a RAR file results in the adapter being available to all J2EE application components running in the server runtime.

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration → Run administrative console**.
3. Log on to the administrative console.
4. Click **Resources → Resource Adapters → Resource adapters**.
5. In the Resource adapters page, click **Install RAR**.

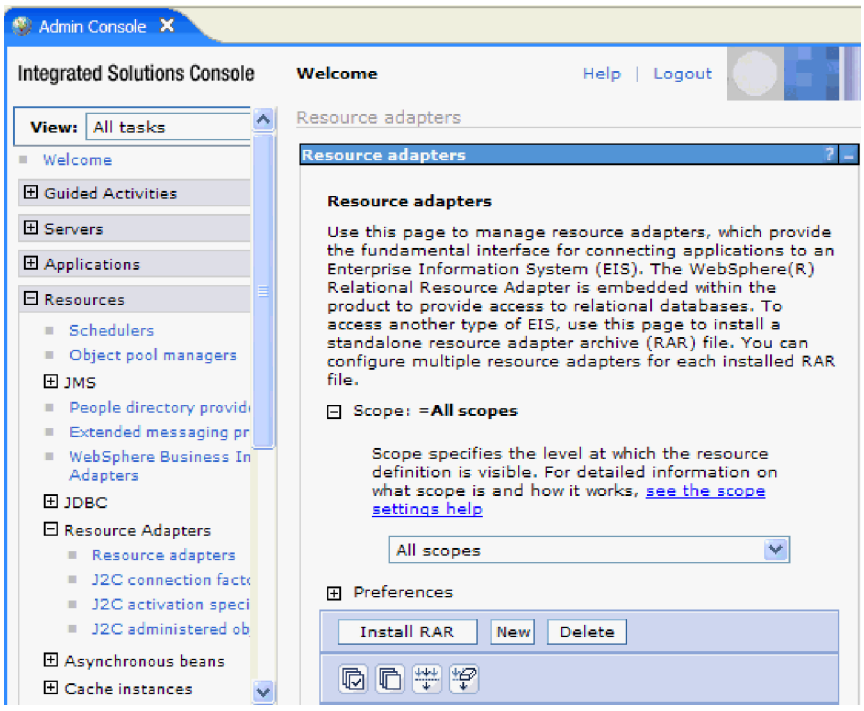


Figure 45. The Install RAR button on the Resource adapters page

6. In the Install RAR file page, click **Browse** and navigate to the RAR file for your adapter.

   The RAR files are typically installed in the following path: *WID_installation_directory*/ResourceAdapters/*adapter_name*/deploy/*adapter*.rar
7. Click **Next**.
8. Optional: In the Resource adapters page, change the name of the adapter and add a description.
9. Click **OK**.
10. Click **Save** in the **Messages** box at the top of the page.

What to do next

The next step is to export the module as an EAR file that you can deploy on the server.

Exporting the module as an EAR file

Using WebSphere Integration Developer, export your module as an EAR file. By creating an EAR file, you capture all of the contents of your module in a format that can be easily deployed to WebSphere Process Server or WebSphere Enterprise Service Bus.

Before you begin

Before you can export a module as an EAR file, you must have created a module to communicate with your service. The module should be displayed in the WebSphere Integration Developer Business Integration perspective.

About this task

To export the module as an EAR file, perform the following procedure.

Procedure

1. Right-click the module and select **Export**.
2. In the Select window, expand **Java EE**.
3. Select **EAR file** and click **Next**.
4. Optional: Select the correct EAR application. The EAR application is named after your module, but with "App" added to the end of the name.
5. Browse for the folder on the local file system where the EAR file will be placed.
6. To export the source files, select the **Export source files** check box. This option is provided in case you want to export the source files in addition to the EAR file. Source files include files associated with Java components, data maps, and so on.
7. To overwrite an existing file, click **Overwrite existing file**.
8. Click **Finish**.

Results

The contents of the module are exported as an EAR file.

What to do next

Install the module in the administrative console. This deploys the module to WebSphere Process Server or WebSphere Enterprise Service Bus.

Installing the EAR file

Installing the EAR file is the last step of the deployment process. When you install the EAR file on the server and run it, the adapter, which is embedded as part of the EAR file, runs as part of the installed application.

Before you begin

You must have exported your module as an EAR file before you can install it on WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

To install the EAR file, perform the following procedure. For more information on clustering adapter module applications, see the http://www.ibm.com/software/webservers/appserv/was/library/.

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration → Run administrative console**.
3. Log on to the administrative console.
4. Click **Applications → New Application → New Enterprise Application**.

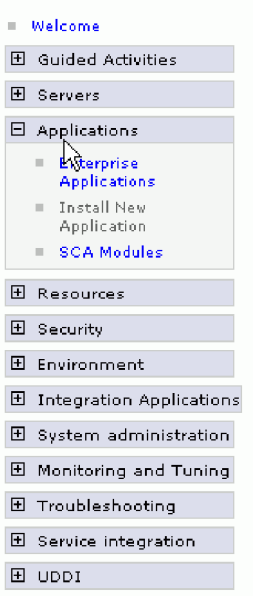


Figure 46. Preparing for the application installation window

5. Click **Browse** to locate your EAR file and click **Next**. The EAR file name is the name of the module followed by "App."
6. Optional: If you are deploying to a clustered environment, complete the following steps.
   a. On the **Step 2: Map modules to servers** window, select the module and click **Next**.
   b. Select the name of the server cluster.
   c. Click **Apply**.
7. Click **Next**. In the Summary page, verify the settings and click **Finish**.
8. Optional: If you are using an authentication alias, complete the following steps:
   a. Expand **Security** and select **Business Integration Security**.
   b. Select the authentication alias that you want to configure. You must have administrator or operator authority to make changes to authentication alias configurations.
   c. Optional: If it is not already filled in, type the **User name**.
   d. If it is not already filled in, type the **Password**.
   e. If it is not already filled in, type the password again in the **Confirm Password** field.
   f. Click **OK**.

Results

The project is now deployed and the Enterprise Applications window is displayed.

What to do next

If you want to set or reset any properties or you would like to cluster adapter project applications, make those changes using the administrative console before configuring troubleshooting tools.

Chapter 7. Administering the adapter module

When you are running the adapter in a stand-alone deployment, use the administrative console of the server to start, stop, monitor, and troubleshoot the adapter module. In an application that uses an embedded adapter, the adapter module starts or stops when the application is started or stopped.

Changing configuration properties for embedded adapters

To change configuration properties after you deploy the adapter as part of a module, you use the administrative console of the runtime environment. You can update resource adapter properties (used for general adapter operation), managed connection factory properties (used for outbound processing), and activation specification properties (used for inbound processing).

Setting resource adapter properties for embedded adapters

To set resource adapter properties for your adapter after it has been deployed as part of a module, use the administrative console. You select the name of the property you want to configure and then change or set the value.

Before you begin

Your adapter module must be deployed on WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

Custom properties are default configuration properties shared by all WebSphere adapters.

To configure properties using the administrative console, use the following procedure.

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration → Run administrative console**.
3. Log on to the administrative console.
4. Select **Applications → Application Types → WebSphere enterprise application**.
5. From the Enterprise Applications list, click the name of the adapter module whose properties you want to change. The **Configuration** page is displayed.

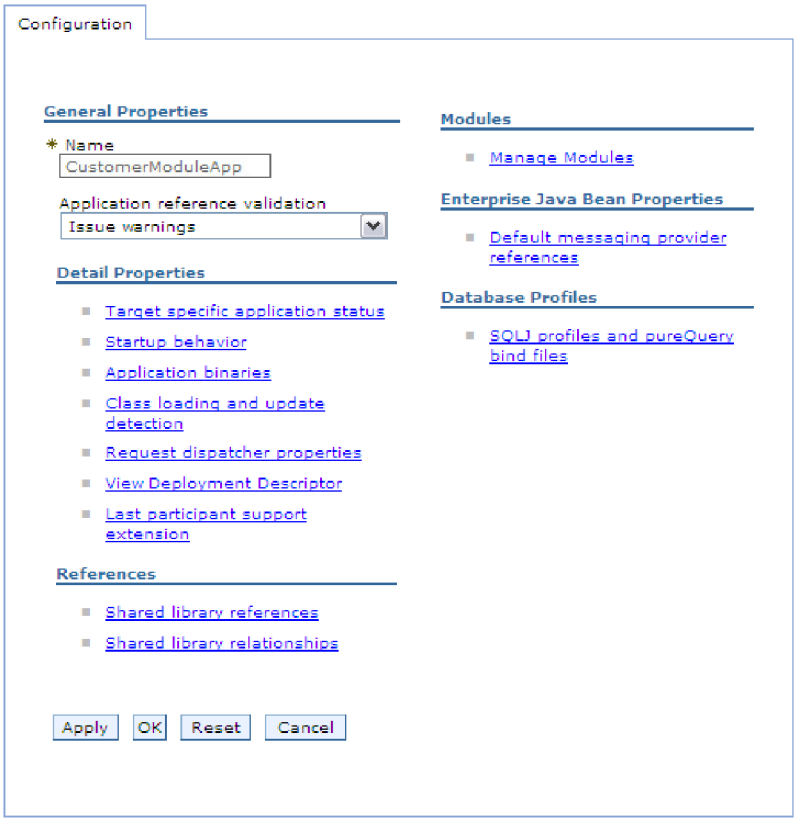


Figure 47. The Manage Modules selection in the Configuration tab

6. Under **Modules**, click **Manage Modules**.
7. Click **IBM WebSphere Adapter for FTP**.
8. From the **Additional Properties** list, click **Resource Adapter**.
9. On the next page, from the **Additional Properties** list, click **Custom properties**.
10. For each property you want to change, perform the following steps.

    Note: See "Resource adapter properties" on page 153 for more information about these properties.

    a. Click the name of the property. The **Configuration** page for the selected property is displayed.
    b. Change the contents of the **Value** field or type a value, if the field is empty.
    c. Click **OK**.
11. In the Messages area, click **Save**.

Results

The resource adapter properties associated with your adapter module are changed.

Setting managed (J2C) connection factory properties for embedded adapters

To set managed connection factory properties for your adapter after it has been deployed as part of a module, use the administrative console. You select the name of the property you want to configure and then change or set the value.

Before you begin

Your adapter module must be deployed on WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

You use managed connection factory properties to configure the target FTP server instance.

Note: In the administrative console, the properties are referred to as "J2C connection factory properties."

To configure properties using the administrative console, use the following procedure.

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration → Run administrative console**.
3. Log on to the administrative console.
4. Select **Applications → Application Types → WebSphere enterprise application**.
5. In the Enterprise Applications list, click the name of the adapter module whose properties you want to change.

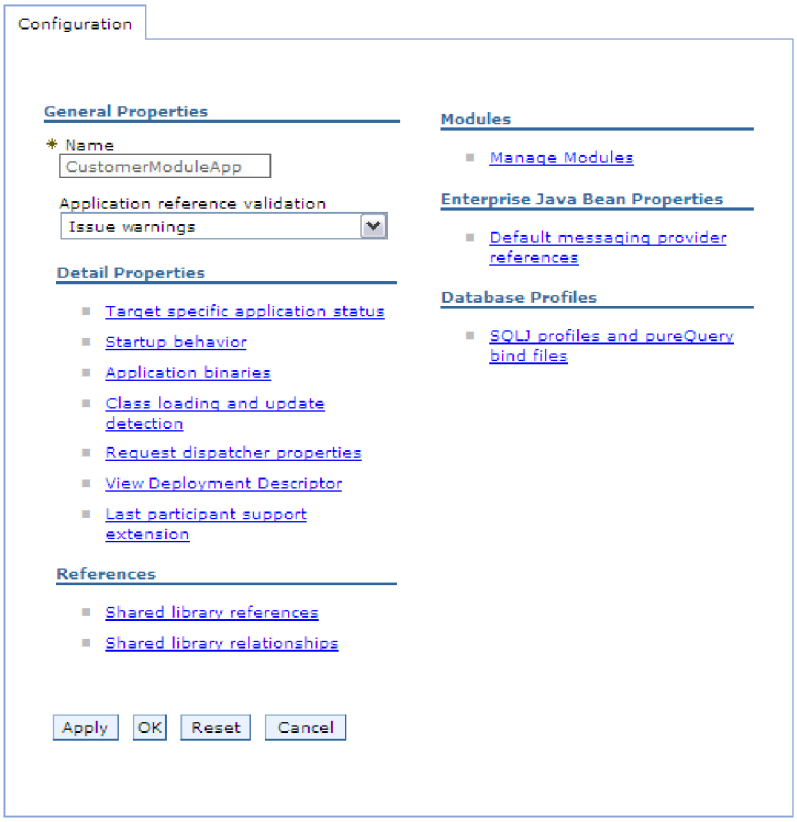


Figure 48. The Manage Modules selection in the Configuration tab

6. Under **Modules**, click **Manage Modules**.
7. Click **IBM WebSphere Adapter for FTP**.
8. In the **Additional Properties** list, click **Resource Adapter**.
9. On the next page, from the **Additional Properties** list, click **J2C connection factories**.
10. Click the name of the connection factory associated with your adapter module.
11. In the **Additional Properties** list, click **Custom properties**.

    Custom properties are those J2C connection factory properties that are unique to Adapter for FTP. Connection pool and advanced connection factory properties are properties you configure if you are developing your own adapter.
12. For each property you want to change, perform the following steps.

    Note: See "Managed (J2C) connection factory properties" on page 158 for more information about these properties.

    a. Click the name of the property.
    b. Change the contents of the **Value** field or type a value, if the field is empty.
    c. Click **OK**.
13. In the Messages area, click **Save**.

Results

The managed connection factory properties associated with your adapter module are changed.

Setting activation specification properties for embedded adapters

To set activation specification properties for your adapter after it has been deployed as part of a module, use the administrative console. You select the name of the message endpoint property you want to configure, and then change or set the value.

Before you begin

Your adapter module must be deployed on WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

You use activation specification properties to configure the endpoint for inbound processing.

To configure properties using the administrative console, use the following procedure.

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration → Run administrative console**.
3. Log on to the administrative console.
4. Select **Applications → Application Types → WebSphere enterprise application**.
5. From the Enterprise Applications list, click the name of the adapter module whose properties you want to change.

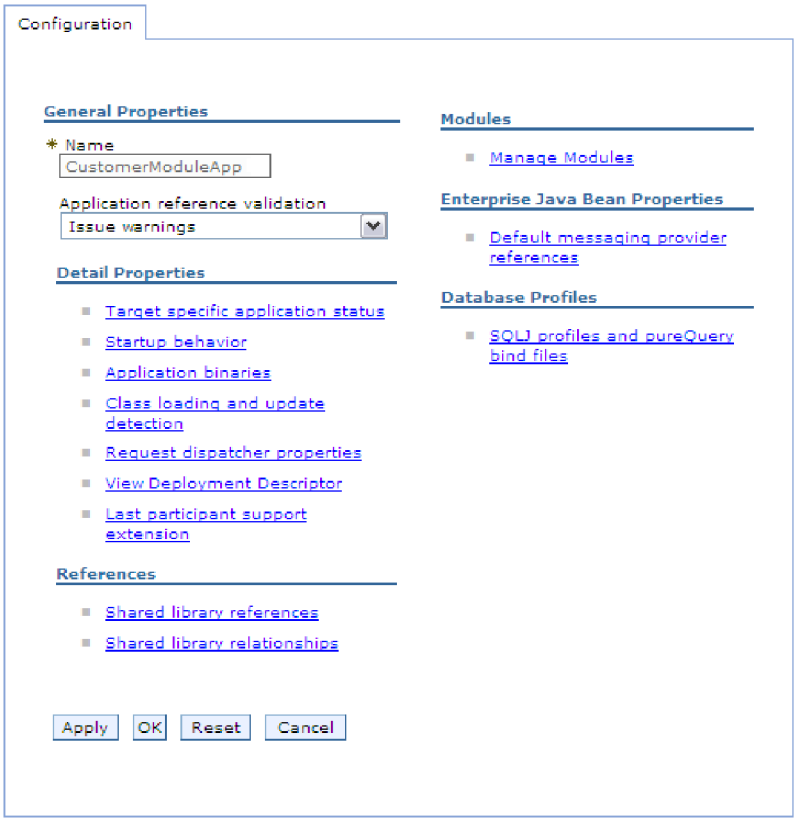


Figure 49. The Manage Modules selection in the Configuration tab

6. Under **Modules**, click **Manage Modules**.
7. Click **IBM WebSphere Adapter for FTP**.
8. From the **Additional Properties** list, click **Resource Adapter**.
9. On the next page, from the **Additional Properties** list, click **J2C activation specifications**.
10. Click the name of the activation specification associated with the adapter module.
11. From the **Additional Properties** list, click **J2C activation specification custom properties**.
12. For each property you want to change, perform the following steps.

    Note: See "Activation specification properties" on page 192 for more information about these properties.

    a. Click the name of the property.
    b. Change the contents of the **Value** field or type a value, if the field is empty.
    c. Click **OK**.
13. In the Messages area, click **Save**.

Results

The activation specification properties associated with your adapter module are changed.

Changing configuration properties for stand-alone adapters

To set configuration properties after you install a stand-alone adapter, you use the administrative console of the runtime environment. You provide general information about the adapter and then set resource adapter properties (which are used for general adapter operation). If the adapter is used for outbound operations, you create a connection factory and then set properties for it. If the adapter is used for inbound operations, you create an activation specification and then set properties for it.

Setting resource adapter properties for stand-alone adapters

To set resource adapter properties for your stand-alone adapter after it has been installed on WebSphere Process Server or WebSphere Enterprise Service Bus, use the administrative console. You select the name of the property you want to configure and then change or set the value.

Before you begin

Your adapter must be installed on WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

Custom properties are default configuration properties shared by all WebSphere adapters.

To configure properties using the administrative console, use the following procedure.

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration → Run administrative console**.
3. Log on to the administrative console.
4. Click **Resources → Resource Adapters → Resource adapters**.
5. In the Resource adapters page, click **IBM WebSphere Adapter for FTP**.
6. In the **Additional Properties** list, click **Custom properties**.
7. For each property you want to change, perform the following steps.

   Note: See “Resource adapter properties” on page 153 for more information about these properties.

   a. Click the name of the property.
   b. Change the contents of the **Value** field or type a value, if the field is empty.
   c. Click **OK**.
8. In the Messages area, click **Save**.

Results

The resource adapter properties associated with your adapter are changed.

Setting managed (J2C) connection factory properties for stand-alone adapters

To set managed connection factory properties for your stand-alone adapter after it has been installed on WebSphere Process Server or WebSphere Enterprise Service Bus, use the administrative console. You select the name of the property you want to configure and then change or set the value.

Before you begin

Your adapter must be installed on WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

You use managed connection factory properties to configure the target FTP server instance.

Note: In the administrative console, the properties are referred to as "J2C connection factory properties."

To configure properties using the administrative console, use the following procedure.

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration → Run administrative console**.
3. Log on to the administrative console.
4. Click **Resources → Resource Adapters → Resource adapters**.
5. In the Resource adapters page, click **IBM WebSphere Adapter for FTP**.
6. In the **Additional Properties** list, click **J2C connection factories**.
7. If you are going to use an existing connection factory, skip ahead to select from the list of existing connection factories.

   Note: If you have selected **Specify connection properties** when you used the external service wizard to configure the adapter module, you do not need to create a connection factory.

   If you are creating a connection factory, perform the following steps:

   a. Click **New**.
   b. In the **General Properties** section of the **Configuration** tab, type a name for the connection factory. For example, you can type `AdapterCF`.
   c. Type a value for **JNDI name**. For example, you can type `com/eis/AdapterCF`.
   d. Optional: Select an authentication alias from the **Component-managed authentication alias** list.
   e. Click **OK**.
   f. In the Messages area, click **Save**.

      The newly created connection factory is displayed.

Figure 50. User defined connection factories for use with the resource adapter

8. In the list of connection factories, click the one you want to use.
9. In the **Additional Properties** list, click **Custom properties**.

   Custom properties are those J2C connection factory properties that are unique to Adapter for FTP. Connection pool and advanced connection factory properties are properties you configure if you are developing your own adapter.
10. For each property you want to change, perform the following steps.

    Note: See "Managed (J2C) connection factory properties" on page 158 for more information about these properties.

    a. Click the name of the property.
    b. Change the contents of the **Value** field or type a value, if the field is empty.
    c. Click **OK**.
11. After you have finished setting properties, click **Apply**.
12. In the Messages area, click **Save**.

Results

The managed connection factory properties associated with your adapter are set.

Setting activation specification properties for stand-alone adapters

To set activation specification properties for your stand-alone adapter after it has been installed on WebSphere Process Server or WebSphere Enterprise Service Bus, use the administrative console. You select the name of the message endpoint property you want to configure, and then change or set the value.

Before you begin

Your adapter must be installed on WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

You use activation specification properties to configure the endpoint for inbound processing.

To configure properties using the administrative console, use the following procedure.

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration → Run administrative console**.
3. Log on to the administrative console.
4. Click **Resources → Resource Adapters → Resource adapters**.
5. In the Resource adapters page, click **IBM WebSphere Adapter for FTP**.
6. In the **Additional Properties** list, click **J2C activation specifications**.
7. If you are going to use an existing activation specification, skip ahead to select from an existing list of activation specifications.

   Note: If you have selected **Use predefined connection properties** when you used the external service wizard to configure the adapter module, you do not need to create an activation specification.

   If you are creating an activation specification, perform the following steps:
   a. Click **New**.
   b. In the **General Properties** section of the **Configuration** tab, type a name for the activation specification. For example, you can type `AdapterAS`.
   c. Type a value for **JNDI name**. For example, you can type `com/eis/AdapterAS`.
   d. Optional: Select an authentication alias from the **Authentication alias** list.
   e. Select a message listener type.
   f. Click **OK**.
   g. Click **Save** in the **Messages** box at the top of the page.

      The newly created activation specification is displayed.
8. In the list of activation specifications, click the one you want to use.
9. In the Additional Properties list, click **J2C activation specification custom properties**.
10. For each property you want to set, perform the following steps.

    Note: See "Activation specification properties" on page 192 for more information about these properties.
    a. Click the name of the property.
    b. Change the contents of the **Value** field or type a value, if the field is empty.
    c. Click **OK**.
11. After you have finished setting properties, click **Apply**.
12. In the Messages area, click **Save**.

Results

The activation specification properties associated with your adapter are set.

Starting the application that uses the adapter

Use the administrative console of the server to start an application that uses the adapter. By default, the application starts automatically when the server starts.

About this task

Use this procedure to start the application, whether it is using an embedded or a stand-alone adapter. For an application that uses an embedded adapter, the adapter starts when the application starts. For an application that uses a stand-alone adapter, the adapter starts when the application server starts.

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration → Run administrative console**.
3. Log on to the administrative console.
4. Click **Applications → Application Types → WebSphere enterprise applications**.

   Note: The administrative console is labeled "Integrated Solutions Console".
5. Select the application that you want to start. The application name is the name of the EAR file you installed, without the .EAR file extension.
6. Click **Start**.

Results

The status of the application changes to Started, and a message stating that the application has started displays at the top of the administrative console.

Stopping the application that uses the adapter

Use the administrative console of the server to stop an application that uses the adapter. By default, the application stops automatically when the server stops.

About this task

Use this procedure to stop the application, whether it is using an embedded or a stand-alone adapter. For an application with an embedded adapter, the adapter stops when the application stops. For an application that uses a stand-alone adapter, the adapter stops when the application server stops.

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration → Run administrative console**.
3. Log on to the administrative console.
4. Click **Applications → Application Types → WebSphere enterprise applications**.

   Note: The administrative console is labeled "Integrated Solutions Console".
5. Select the application that you want to stop. The application name is the name of the EAR file you installed, without the .EAR file extension.
6. Click **Stop**.

Results

The status of the application changes to Stopped, and a message stating that the application has stopped displays at the top of the administrative console.

Monitoring performance using Performance Monitoring Infrastructure

Performance Monitoring Infrastructure (PMI) is a feature of the administrative console that allows you to dynamically monitor the performance of components in the production environment, including the adapter for FTP. PMI collects adapter performance data, such as average response time and total number of requests, from various components in the server and organizes the data into a tree structure. You can view the data through the Tivoli® Performance Viewer, a graphical monitoring tool that is integrated with the administrative console in WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

You can monitor the performance of your adapter by having PMI collect data at the following points:

- At outbound processing to monitor outbound requests
- At inbound event retrieval to monitor the retrieval of an event from the event table
- At inbound event delivery to monitor the delivery of an event to the endpoint or endpoints

Before you can enable and configure PMI for your adapter, you must first set the level of tracing detail and run some events from which to gather performance data.

To learn more about how PMI can help you monitor and improve the overall performance of your adapter environment, search for PMI on the WebSphere Application Server web site: http://www.ibm.com/software/webservers/appserv/was/library/.

Configuring Performance Monitoring Infrastructure

You can configure Performance Monitoring Infrastructure (PMI) to gather adapter performance data, such as average response time and total number of requests. After you configure PMI for your adapter, you can monitor the adapter performance using Tivoli Performance viewer.

Before you begin

Before you can configure PMI for your adapter, you must first set the level of tracing detail and run some events from which to gather performance data.

1. To enable tracing and to receive event data, the trace level must be set to either fine, finer, finest, or all. After *=info, add a colon and a string, for example:

   ```
   *=info: WBILocationMonitor.CEI.ResourceAdapter.
   *=finest: WBILocationMonitor.LOG.ResourceAdapter.*=finest:
   ```

   For instructions on setting the trace level, refer to “Enabling tracing with the Common Event Infrastructure (CEI)” on page 132.
2. Generate at least one outbound request or inbound event to produce performance data that you can configure.

Procedure

1. Enable PMI for your adapter.
   a. In the administrative console, expand **Monitoring and Tuning**, and then select **Performance Monitoring Infrastructure (PMI)**.
   b. From the list of servers, click the name of your server.
   c. Select the Configuration tab, and then select the **Enable Performance Monitoring (PMI)** check box.
   d. Select **Custom** to selectively enable or disable statistics.

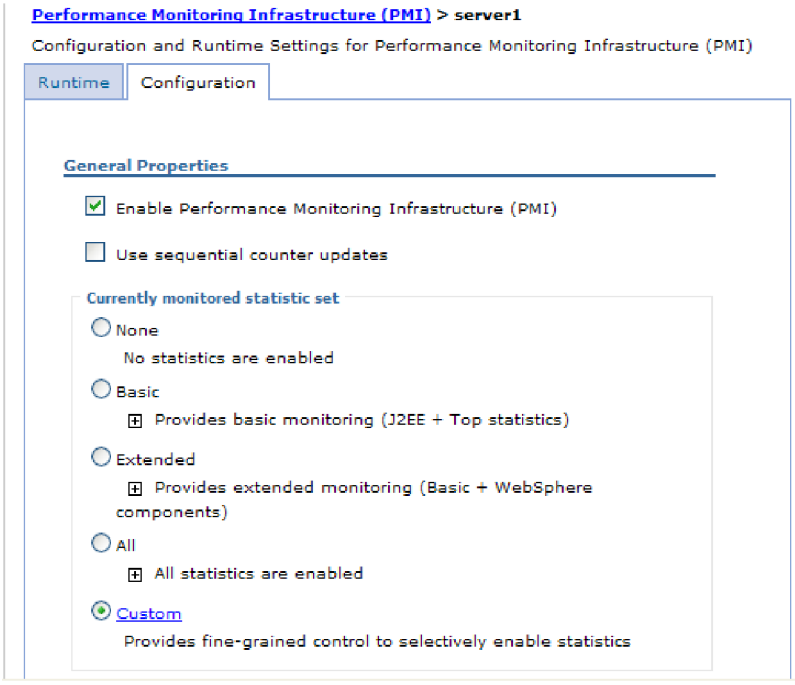


Figure 51. Enabling Performance Monitoring Infrastructure

   e. Click **Apply** or **OK**.
   f. Click **Save**. PMI is now enabled.
2. Configure PMI for your adapter.
   a. In the administrative console, expand **Monitoring and Tuning**, and then select **Performance Monitoring Infrastructure (PMI)**.
   b. From the list of servers, click the name of your server.
   c. Select **Custom**.
   d. Select the **Runtime** tab. The following figure shows the Runtime tab.

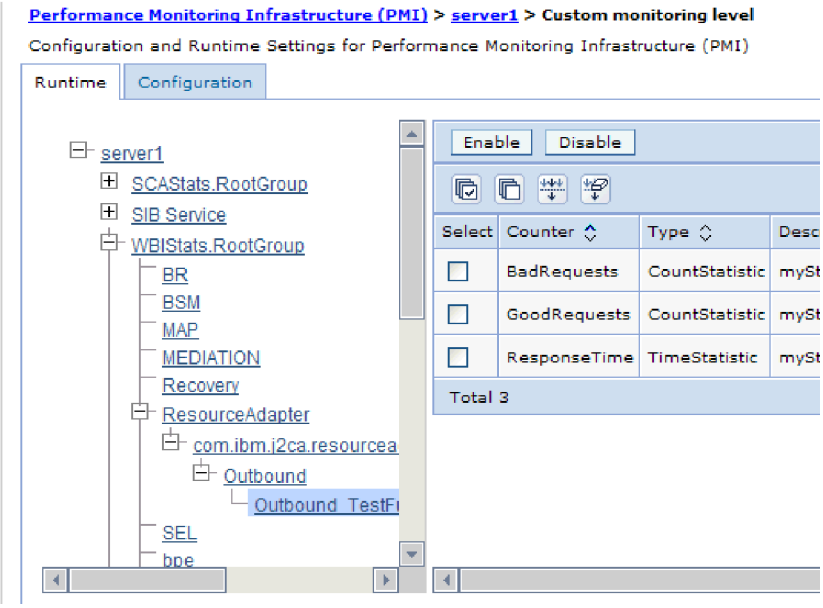


Figure 52. Runtime tab used for configuring PMI

e. Click **WBIStats.RootGroup**. This is a PMI sub module for data collected in the root group. This example uses the name WBIStats for the root group.

f. Click **ResourceAdapter**. This is a sub module for the data collected for the JCA adapters.

g. Click the name of your adapter, and select the processes you want to monitor.

h. In the right pane, select the check boxes for the statistics you want to gather, and then click **Enable**.

Results

PMI is configured for your adapter.

What to do next

Now you can view the performance statistics for your adapter.

Enabling tracing with the Common Event Infrastructure (CEI)

The adapter can use the Common Event Infrastructure, a component embedded in the server, to report data about critical business events such as the starting or stopping of a poll cycle. Event data can be written to a database or a trace log file depending on configuration settings.

About this task

Procedure

1. In the administrative console, click **Troubleshooting**.
2. Click **Logs and Trace**.
3. In the list of servers, click the name of your server.

4. In the **Change Log Detail Levels** box, click the name of the CEI database (for example, WBIEventMonitor.CEI.ResourceAdapter.*) or the trace log file (for example, WBIEventMonitor.LOG.ResourceAdapter.*) to which you want the adapter to write event data.
5. Select the level of detail about business events that you want the adapter to write to the database or trace log file, and (optionally) adjust the granularity of detail associated with messages and traces.
   - **No Logging**. Turns off event logging.
   - **Messages Only**. The adapter reports an event.
   - **All Messages and Traces**. The adapter reports details about an event.
   - **Message and Trace Levels**. Settings for controlling the degree of detail the adapter reports about the business object payload associated with an event. If you want to adjust the detail level, select one of the following options:

     Fine. The adapter reports the event but none of the business object payload.

     Finer. The adapter reports the event and the business object payload description.

     Finest. The adapter reports the event and the entire business object payload.
6. Click **OK**.

Results

Event logging is enabled. You can view CEI entries in the trace log file or by using the Common Base Event Browser within the administrative console.

Viewing performance statistics

You can view adapter performance data through the graphical monitoring tool, Tivoli Performance Viewer. Tivoli Performance Viewer is integrated with the administrative console in WebSphere Process Server or WebSphere Enterprise Service Bus.

Before you begin

Configure Performance Monitoring Infrastructure for your adapter.

Procedure

1. In the administrative console, expand **Monitoring and Tuning**, expand **Performance Viewer**, then select **Current Activity**.
2. In the list of servers, click the name of your server.
3. Under your server name, expand **Performance Modules**.
4. Click **WBIStatsRootGroup**.
5. Click **ResourceAdapter** and the name of your adapter module.
6. If there is more than one process, select the check boxes for the processes whose statistics you want to view.

Results

The statistics are displayed in the right panel. You can click **View Graph** to view a graph of the data, or **View Table** to see the statistics in a table format.

The following figure shows adapter performance statistics.

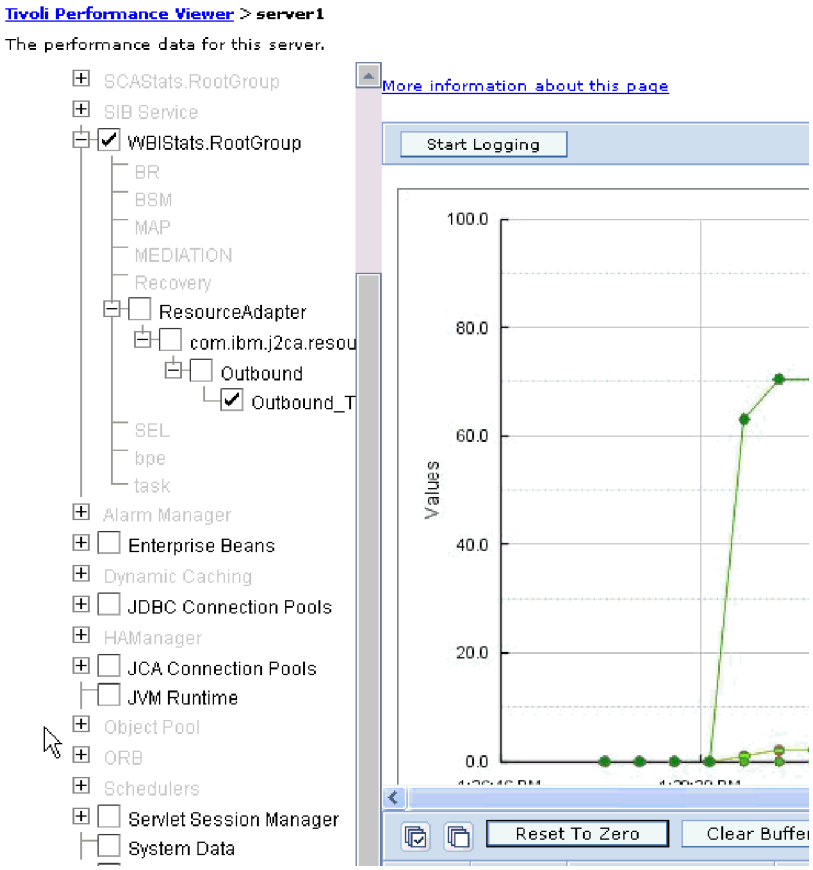


Figure 53. Adapter performance statistics, using graph view

Chapter 8. Troubleshooting and support

Common troubleshooting techniques and self-help information help you identify and solve problems quickly.

ServerToServerFileTransfer

The specified file is transferred from one FTP server directory to another FTP server directory.

If the value of the host name property is set to `localhost`, the first server is located on the same system as the adapter workstation. Adapter for FTP generates the following error: 421 error-Can't open data connection. To work around this problem, edit the hosts file (which, for the Windows platform, is located at `<WindowsHome>/system32/drivers/etc/hosts`) and add a new entry with the external IP address, for example, 9.186.116.151 localhost.

The adapter will also work if the host name value or the external IP address is used for example, if FTPTEST is used as the host name format or 9.186.116.151 is used as the IP format.

Configuring logging and tracing

Configure logging and tracing to suit your requirements. Enable logging for the adapter to control the status of event processing. Change the adapter log and trace file names to separate them from other log and trace files.

Configuring logging properties

Use the administrative console to enable logging and to set the output properties for a log, including the location, level of detail, and output format of the log.

About this task

Before the adapters can log monitored events, you must specify the service component event points that you want to monitor, what level of detail you require for each event, and format of the output used to publish the events to the logs. Use the administrative console to perform the following tasks:

- Enable or disable a particular event log
- Specify the level of detail in a log
- Specify where log files are stored and how many log files are kept
- Specify the format for log output

  If you set the output for log analyzer format, you can open trace output using the Log Analyzer tool, which is an application included with your process server. This is useful if you are trying to correlate traces from two different server processes, because it allows you to use the merge capability of the Log Analyzer.

For more information about monitoring on a process server, including service components and event points, see the documentation for your process server.

You can change the log configuration statically or dynamically. Static configuration takes effect when you start or restart the application server. Dynamic, or run time, configuration changes apply immediately.

When a log is created, the detail level for that log is set from the configuration data. If no configuration data is available for a particular log name, the level for that log is obtained from the parent of the log. If no configuration data exists for the parent log, the parent of that log is checked, and so on, up the tree, until a log with a non-null level value is found. When you change the level of a log, the change is propagated to the children of the log, which recursively propagate the change to their children, as necessary.

To enable logging and set the output properties for a log, use the following procedure.

Procedure

1. In the navigation pane of the administrative console, click **Servers** → **Application Servers**.
2. Click the name of the server that you want to work with.
3. Under **Troubleshooting**, click **Logs and trace**.
4. Click **Change Log Detail Levels**.
5. Specify when you want the change to take effect:
   - For a static change to the configuration, click the **Configuration** tab.
   - For a dynamic change to the configuration, click the **Runtime** tab.
6. Click the names of the packages whose logging level you want to modify. The package names for WebSphere Adapters start with **com.ibm.j2ca.***:
   - For the adapter base component, select **com.ibm.j2ca.base.***.
   - For the adapter base component and all deployed adapters, select **com.ibm.j2ca.***.
   - For the Adapter for FTP only, select the **com.ibm.j2ca.ftp.*** package.
7. Select the logging level.

Logging Level	Description
Fatal	The task cannot continue or the component cannot function.
Severe	The task cannot continue, but the component can still function. This logging level also includes conditions that indicate an impending fatal error, that is, situations that strongly suggest that resources are on the verge of being depleted.
Warning	A potential error has occurred or a severe error is impending. This logging level also includes conditions that indicate a progressive failure, for example, the potential leaking of resources.
Audit	A significant event has occurred that affects the server state or resources.
Info	The task is running. This logging level includes general information outlining the overall progress of a task.
Config	The status of a configuration is reported or a configuration change has occurred.
Detail	The subtask is running. This logging level includes general information detailing the progress of a subtask.

8. Click **Apply**.

9. Click **OK**.
10. To have static configuration changes take effect, stop and then restart the process server.

Results

Log entries from this point forward contain the specified level of information for the selected adapter components.

Changing the log and trace file names

To keep the adapter log and trace information separate from other processes, use the administrative console to change the file names. By default, log and trace information for all processes and applications on a process server is written to the `SystemOut.log` and `trace.log` files.

Before you begin

You can change the log and trace file names at any time after the adapter module has been deployed to an application server.

About this task

You can change the log and trace file names statically or dynamically. Static changes take effect when you start or restart the application server. Dynamic or run time changes apply immediately.

Log and trace files are in the *install_root*`/profiles/`*profile_name*`/logs/`*server_name* folder.

To set or change the log and trace file names, use the following procedure.

Procedure

1. In the navigation pane of the administrative console, select **Applications > Enterprise Applications**.
2. In the Enterprise Applications list, click the name of the adapter application. This is the name of the EAR file for the adapter, but without the ear file extension. For example, if the EAR file is named `Accounting_OutboundApp.ear`, then click **Accounting_OutboundApp**.
3. In the Configuration tab, in the Modules list, click **Manage Modules**.
4. In the list of modules, click IBM WebSphere Adapter for FTP.
5. In the Configuration tab, under Additional Properties, click **Resource Adapter**.
6. In the Configuration tab, under Additional Properties, click **Custom properties**.
7. In the Custom Properties table, change the file names.
   a. Click either **logFilename** to change the name of the log file or **traceFilename** to change the name of the trace file.
   b. In the Configuration tab, type the new name in the **Value** field. By default, the log file is called `SystemOut.log` and the trace file is called `trace.log`.
   c. Click **Apply** or **OK**. Your changes are saved on your local machine.
   d. To save your changes to the master configuration on the server, use one of the following procedures:

- **Static change**: Stop and restart the server. This method allows you to make changes, but those changes do not take effect until you stop and start the server.
- **Dynamic change**: Click the **Save** link in the Messages box above the Custom properties table. Click **Save** again when prompted.

Known issues in editing the Rule Table

When configuring the adapter to filter event files based on a set of rules, some known issues can occur while editing the Rule Table in the Properties view. To correct the problem follow the solutions described here for each of these issues.

Symptoms:

When an existing Rule Table row is configured in the Properties view, the following issue can occur:

The **Finish** option is not enabled sometimes.

Problem:

After you have completed entering all the required properties, the **Finish** option is not enabled for you to complete the editing of the Rule Table.

Solution:

To correct this problem, use either of the following workaround:

1. Use **Tab** to move between the fields.
2. Keep the focus away from the **Value** field either to **Operator** or the **Property** field.

Support for global elements without wrapper

When global element without wrapper is used as input type, you need to take care of using the correct configuration described for the below listed scenarios to get the expected result.

Global element of named type without wrapper during outbound processing

When global element of named type without wrapper is used as input type in adapter outbound using UTF8XML Datahandler, the file is serialized with global element type name as root element name, instead of the global element name.

To serialize file to get the global element name as the root element name, you need to use the XML Datahandler and specify the global element name as the root element name in XML datahandler configuration.

Global element of anonymous type without wrapper

When global element of anonymous type without wrapper is used as input type in adapter inbound or outbound retrieve, the data object is emitted back to SCA component. When this data object is serialized, it returns the type name of dataobject as ‘globalelementname_._type'.

To get the correct data object type, in order to be used for a global element of anonymous type without wrapper, for inbound as well as outbound retrieve, you need to use the following code snippet.

```
The following sample code can be used to get the correct
dataobject details for global element of anonymous type
without wrapper, which is named as GlobalElementExample1.

import java.io.ByteArrayOutputStream;
import java.io.IOException;

import commonj.sdo.DataObject;
import commonj.sdo.Type;

import com.ibm.websphere.bo.BOFactory;
import com.ibm.websphere.bo.BOXMLSerializer;
import com.ibm.websphere.sca.ServiceManager;

public void emit(DataObject globalElementExample1) {
ServiceManager s = ServiceManager.INSTANCE;
BOFactory factory= (BOFactory) s.locateService
("com/ibm/websphere/bo/BOFactory");
DataObject dobj= factory.createByElement
(globalElementExample1.getType().getURI(), "GlobalElementExample1");
final Type type = dobj.getType();
String typeName = type.getName();
if (typeName.endsWith("_._type"))
         typeName = typeName.substring(0, typeName.indexOf("_._type"));
BOXMLSerializer serializer = BOXMLSerializer)s.locateService
("com/ibm/websphere/bo/BOXMLSerializer");
ByteArrayOutputStream baos = new ByteArrayOutputStream();
serializer.writeDataObject(globalElementExample1, type.getURI(), typeName, baos);
String bo = new String(baos.toByteArray());
System.out.println("bo : "+bo);
}
```

First-failure data capture (FFDC) support

The adapter supports first-failure data capture (FFDC), which provides persistent records of failures and significant software incidents that occur during run time in WebSphere Process Server or WebSphere Enterprise Service Bus.

The FFDC feature runs in the background and collects events and errors that occur at run time. The feature provides a means for associating failures to one another, allowing software to link the effects of a failure to their causes, and thereby facilitate the quick location of the root cause of a failure. The data that is captured can be used to identify exception processing that occurred during the adapter run time.

When a problem occurs, the adapter writes exception messages and context data to a log file, which is located in the *install_root*/profiles/*profile*/logs/ffdc directory.

For more information about first-failure data capture (FFDC), see the WebSphere Process Server or WebSphere Enterprise Service Bus documentation.

org.xml.sax.SAXParseException

When the adapter is configured with the XML data handler, an org.xml.sax.SAXParseException exception is generated if the content is not in the specified business object format. To correct the problem, make sure the file content matches the business object structure. If the file contains multiple business objects, make sure the delimiter is specified correctly.

Symptom:

When the adapter is configured with the XML data handler, the following exception is thrown:

org.xml.sax.SAXParseException: Content is not allowed in trailing section

Problem:

The content of the file is not in the specified business object format.

Solution:

To correct this problem, use the following procedure:

1. Make sure the file content matches the business object structure.
2. If the content file contains multiple business objects, make sure the delimiter is specified correctly.

Self-help resources

Use the resources of IBM software support to get the most current support information, obtain technical documentation, download support tools and fixes, and avoid problems with WebSphere Adapters. The self-help resources also help you diagnose problems with the adapter and provide information about how to contact IBM software support.

Support Web site

The WebSphere Adapters software support Web site at http://www.ibm.com/software/integration/wbiadapters/support/ provides links to many resources to help you learn about, use, and troubleshoot WebSphere Adapters, including:

- Flashes (alerts about the product)
- Technical information including the product information center, manuals, IBM Redbooks®, and whitepapers
- Educational offerings
- Technotes

Recommended fixes

A list of recommended fixes you must apply is available at the following location: http://www.ibm.com/support/docview.wss?fdoc=aimadp&rs=695&uid=swg27010397

Technotes

Technotes provide the most current documentation about the Adapter for FTP, including the following topics:

- Problems and their currently available solutions
- Answers to frequently asked questions
- How to information about installing, configuring, using, and troubleshooting the adapter
- *IBM Software Support Handbook*

For a list of technotes for WebSphere Adapters, visit this address:

http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8&dc=DB520+D800+D900+DA900+DA800+DB560&dtm

Plug-in for IBM Support Assistant

Adapter for FTP provides a plug-in for IBM Support Assistant, which is a free, local software serviceability workbench. The plug-in supports the dynamic trace feature. For information about installing or using IBM Support Assistant, visit this address:

http://www.ibm.com/software/support/isa/

Chapter 9. Reference information

To support you in your tasks, reference information includes details about business objects that are generated by the external service wizard and information about adapter properties, including those that support bidirectional transformation. It also includes pointers to adapter messages and related product information.

Business object information

You can determine the purpose of a business object by examining both the application-specific information within the business object definition file and the name of the business object. The application-specific information dictates what operations can be performed on the FTP server. The name typically reflects the operation to be performed and the structure of the business object.

Business object structure

The adapter supports three different types of business object structures. A generic business object, which is used to pass unstructured data. A generic business object with a business graph, which contains the action to be performed on the data and the connection-specific information. A user-defined type, which is a content-specific business object that supports specific business object structures (such as customer and order business objects).

Business graphs are optional and can be selected in the external service wizard.

The FTPFileBG, FTPFile, and UnstructuredContent generic business object definitions are automatically generated. Depending on the custom complex types selected when you create external services, the corresponding business object or objects definitions will also be generated. For example, if you select Customer, including the optional business graph, the CustomerWrapperBG and CustomerWrapper business objects will be generated.

FTPFileBG

The FTPFileBG business object is a generic business object that contains the verb (the action to be performed on the data) and the FTPFile business object as a child. The following graphic illustrates this relationship.

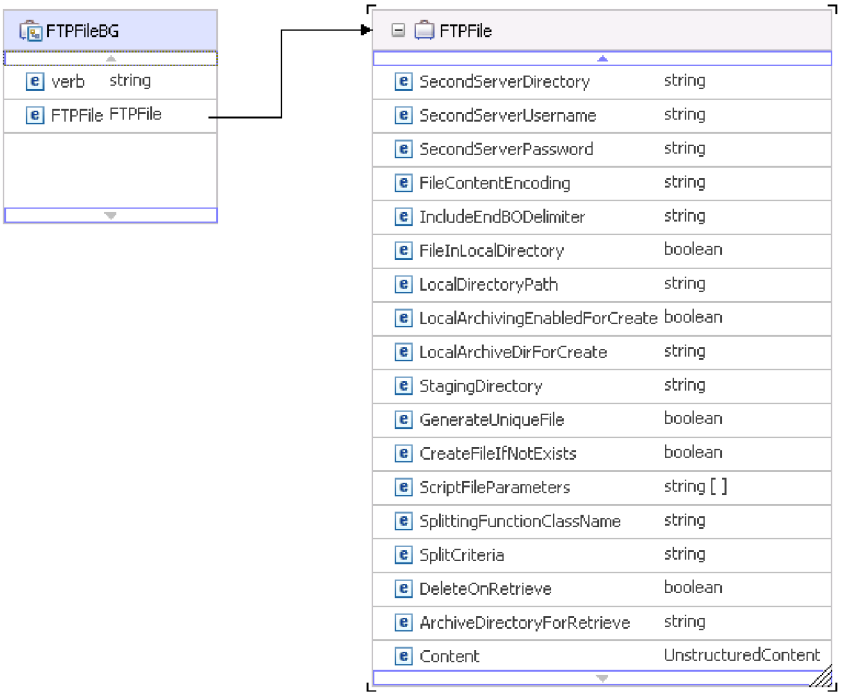


Figure 54. FTPFileBG business object

FTPFile

The FTPFile business object contains all necessary connection information, and an UnstructuredContent business object as a child. The following graphic illustrates this relationship.

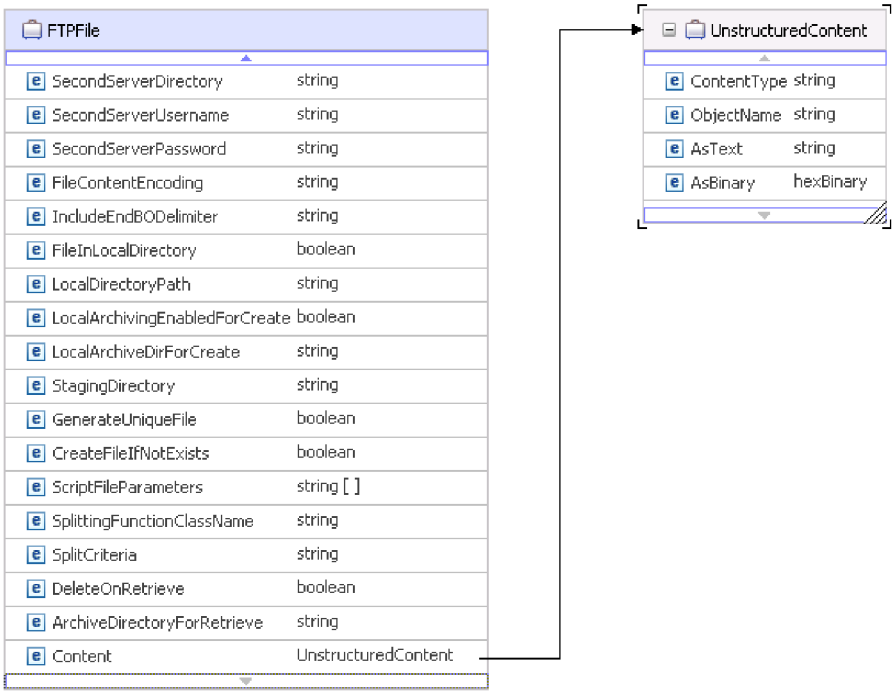


Figure 55. FTPFile business object

CustomerWrapperBG

The CustomerWrapperBG is a business object that contains the verb (the action to be performed on the data) and the CustomerWrapper business object as a child. The following graphic illustrates this relationship.

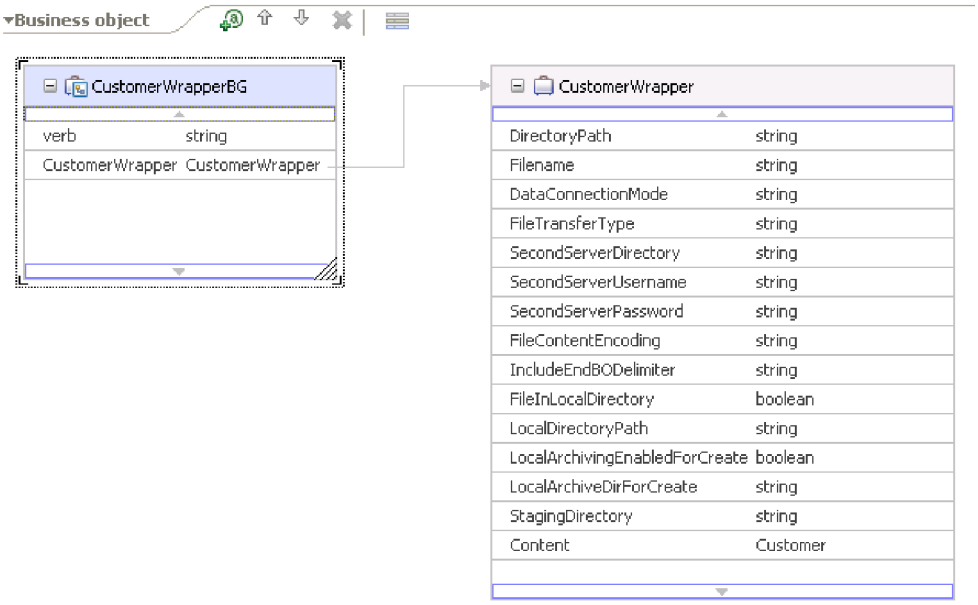


Figure 56. CustomerWrapperBG business object

CustomerWrapper

The CustomerWrapper business object is a business object that contains all necessary connection information and the content-specific Customer business object as a child. The following graphic illustrates this relationship.

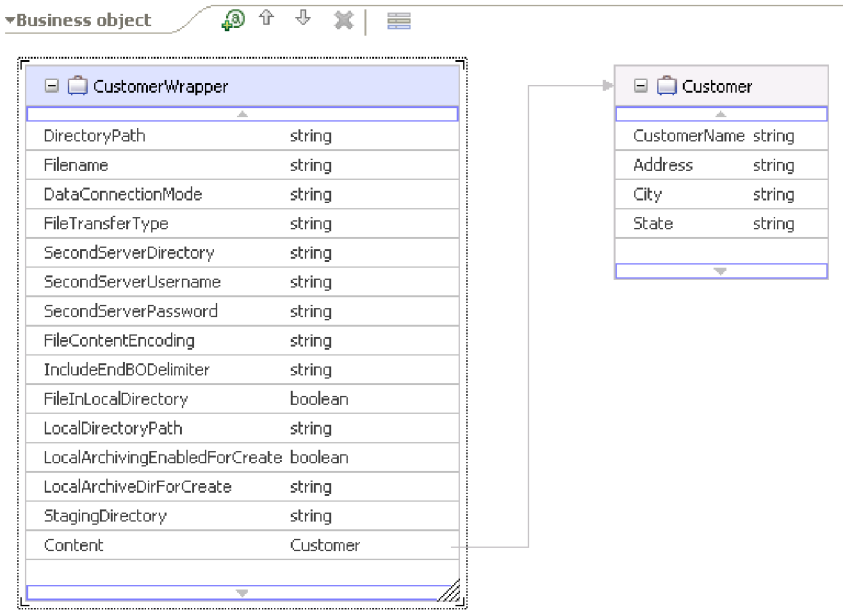


Figure 57. CustomerWrapper business object

Global elements in a structured business object

The Adapter for FTP supports global elements in structured business objects. Global elements with null namespace are also supported.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema elementFormDefault="qualified"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ftp="http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/customer"
            targetNamespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/customer">

<xsd:element name="CustomerType1">
    <xsd:complexType>
        <xsd:sequence>
               <xsd:element name="name" type="xsd:string"></xsd:element>
                 <xsd:element name="address" type="xsd:string"></xsd:element>
                 <xsd:element name="city" type="xsd:string"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>
</xsd:schema>
```

Figure 58. Structure of the global elements in a structured Business Object

The CustomerType1 is the global element in the above business object.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema elementFormDefault="qualified"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ftp="http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/customer"
            targetNamespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/customer">

<xsd:element name="CustomerInventory" type="ftp:CustomerInventoryType3"/>

<xsd:complexType name="CustomerInventoryType3">
    <xsd:sequence>
      <xsd:element name="shipTo" type="xsd:string"/>
      <xsd:element name="billTo" type="xsd:string"/>
      <xsd:element name="items"  type="xsd:string"/>
    </xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

The CustomerInventory is the global element in the above business object.

Naming conventions

When the external service wizard generates a business object, it provides a name for the business object based on the name of the object in the FTP server that it uses to build the business object. Use the Business Object Editor to create user-defined objects.

External service wizard converts the name of the object to mixed case, which means that it removes any separators, such as spaces or underscores, and then capitalizes the first letter of each word. For example, if the external service wizard uses a FTP server object called CUSTOMER_ADDRESS to generate a business object, it generates a business object called CustomerAddress.

The generated business object name can indicate the structure of the business object. However, business objects names have no semantic value to the adapter. This means that if you change the business object name, the behavior of the business object remains the same.

Important: If you choose to rename a business object, use the refactoring functionality in WebSphere Integration Developer to ensure that you update all the business object dependencies. For instructions on using refactoring to rename business objects, refer to the following link: http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/topic/com.ibm.wbit.help.brules.doc/selector/topics/trefacts.html.

Note: Business graph generation is optional and is supported for WebSphere Process Server only.

Support for null namespace

FTP adapter supports the business objects with null namespaces.

FTP adapter supports the business objects with null namespaces. You can configure the namespace value of the business object using the Business Object Editor tool which is provided by the WebSphere Integration Developer (WID).

Note: Use the default value provided by the WebSphere Integration Developer, for example, http://ModuleName and configure the null namespaces. If a business object is created using the tool, the default namespace will be set as the module name. This can be modified with any other value or set to null.

Business object attribute properties

Business object architecture defines various properties that apply to attributes. This section describes how the adapter interprets these properties.

The following table describes these properties and how the adapter interprets them.

Table 8. Business object attribute properties

Property	Description
Cardinality	For simple attributes, 1 is used. For container attributes, depending on the method requirements, `n` is used.
Foreign Key	The adapter does not have any specific elements representing Foreign Keys.
Key	The adapter does not have any specific elements representing a Key.
Name	This property represents the unique name of the attribute, if it is a simple attribute, or the name of the business object, if it is a child business object.
Required	This property specifies whether an attribute must contain a value.
Type	The attribute type can be either simple or complex. Simple types are: Boolean, String, LongText, Integer, Float, Double and Byte[]. A typical complex type is the name of another business object.

Business object operation support

An operation is the name of the action that is performed on the business object by the adapter. Every business object has an operation associated with it. The name of the operation typically indicates the type of action that is taken on the business object.

The following table defines the operations that the adapter supports.

Table 9. Supported operations of business objects

Operation	Result
Create	Creates a file with the specified file name in the specified directory with the content sent across in the request.
Append	Appends the content in the request to the end of the file.
Retrieve	Returns the content of the file specified in the request.
Delete	Deletes the file from the directory specified in the request.
Overwrite	Overwrites the file in the directory with the content specified in the request.
Exists	Returns a successful response if the file in the request exists in the specified directory or sub directories.
List	Returns all the file names in the specified directory.
ServerToServerFileTransfer	Transfers the file from one FTP server to another FTP server.
ExecuteFTPScript	Runs an FTP script file in the specified directory.

Custom business objects

If you use custom business objects, you must create predefined business objects using WebSphere Integration Developer business object wizard before running the external service wizard. The business object definitions created by the wizard are stored as xsd files on your local system. When the external service wizard creates

business objects, it looks for the predefined business objects created in the business object wizard and populates them with data specific to the module.

For more information about how to create predefined business objects, refer to the WebSphere Integration Developer documentation.

Custom file splitting

You can implement a custom class containing the splitting logic. The adapter provides a Java™ interface for the class. The details of the interface are shown below.

```
public interface SplittingFunctionalityInterface extends Iterator{
  public int getTotalBOs(String filename) throws SplittingException;
public void setBODetails(String filename, int currentPosition, int totalBOs,
  boolean includeEndBODelimiter) throws SplittingException;
   public void setSplitCriteria(String splitCriteria);
   public void setEncoding(String encoding);
   public void setLogUtils(LogUtils logUtils);
public boolean isSplitBySize()
}
```

- `public int getTotalBOs(String filename) throws SplittingException`

  This method returns the total number of business object's present in the event file given by `filename`.

- `public void setSplitCriteria(String splitCriteria)`

  This method takes the `splitCriteria`, which is based on the number of business object's in the event file. Each business object is returned during the `next()` call.

- `public void setLogUtils(LogUtils logUtils)`

  This method is used to set the `LogUtils` object, which is the class that the user can use to write trace and log messages to the files.

- `public void setEncoding(String encoding)`

  This method is used to set the encoding of the event file content. This encoding is used while reading the file content. This encoding is also used for the `SplitCriteria`.

- `public void setBODetails(String filename, int currentPosition, int totalBOs, boolean includeEndBODelimiter) throws SplittingException`

  This method is used to set the current business object number so that whenever a `next()` call is made, the business object number set in the `currentPosition` is returned. It also takes an `includeEndBODelimiter` parameter, which when set to `true`, includes the `SplitCriteria` at the end of the business object content. This method must be called before every `next()` call so that the `next()` method returns the business object content for the business object set in this method.

- The iterator has 3 methods: `hasNext()`, `next` and `remove()`, which also need to be implemented. The `next()` method returns the business object content (as a byte[]) for the business object position set in `setBODetails()`. If the business object position is not set, it fails. The `hasNext()` method indicates whether the business object position set in the `setBODetails()` exists or not. Before a `hasNext()` call, the `setBODetails()` method must be called. The `remove()` method is called for each of the business object entries being deleted from the Event persistence table. Do not delete the event file in this method. Only clean up resources that are being used.

- `public boolean isSplitBySize()`

  This method indicates whether the event file is parsed based on size or based on delimiter.

Fault business objects

The adapter supports business faults, which are exceptions that are anticipated and declared in the outbound service description, or import. Business faults occur at predictable points in a business process, and are caused by a business rule violation or a constraint violation.

The adapter provides the following fault business objects that the wizard creates:

- DuplicateRecordFault

  The adapter generates this fault for the:

  – outbound Create operation when an error occurs because the file specified already exists in the specified directory

  – ServerToServerFileTransfer operation when the file already exists in the second server directory

  – Retrieve operation when the file to be retrieved already exists in the local directory. This occurs when the FileInLocalDirectory property is set to true or when the splitting is enabled.

- RecordNotFoundFault

  The adapter generates this fault when processing the Create, Append, Delete, Overwrite, Retrieve, ExecuteFTPScript, and ServerToServerFileTransfer operations when the file directory path or script file does not exist in the specified directory path. This fault occurs when the directory path does not exist and when the sequence file does not exist during the Create operation.

- MissingDataFault

  The adapter generates this fault when required values are not provided, such as when the file content is null or the file name or directory path is empty.

  During a Retrieve operation, the adapter generates this fault when an error occurs because the delimiter is null or not valid. If splitCriteria is null or invalid when Splittingfunctionclassname is SplitByDelimiter and when LocalDirectoryPath is null, a MissingData fault is thrown with the message that the LocalDirectoryPath is missing. The adapter does not throw an exception when splitCriteria is null or not valid and SplitBySize is configured. During a Retrieve operation, the adapter generates this fault when an error occurs because the delimiter is null and SplitByDelimiter is configured.

Outbound configuration properties

WebSphere Adapter for FTP has several categories of outbound connection configuration properties, which you set with the external service wizard while generating or creating objects and services. You can change the resource adapter and managed connection factory properties after you deploy the module to WebSphere Process Server or WebSphere Enterprise Service Bus using WebSphere Integration Developer or the administrative console, but connection properties for the external service wizard cannot be changed after deployment.

Guide to information about properties

The properties used to configure WebSphere Adapter for FTP are described in detail in tables included in each of the configuration properties topics, such as Resource adapter properties, Managed connection factory properties, and so on. To help you use these tables, information about each row you might see is explained here.

The following table explains the meaning of each row that might be displayed in the table for a configuration property.

Row	Explanation
Required	A required field (property) must have a value in order for the adapter to work. Sometimes the external service wizard provides a default value for required properties. Removing a default value from a required field on the external service wizard *will not change that default value*. When a required field contains no value at all, the external service wizard processes the field using its assigned default value, and that default value is displayed on the administrative console. Possible values are **Yes** and **No**. Sometimes a property is required only when another property has a specific value. When this is the case, the table will note this dependency. For example, • Yes, when the EventQueryType property is set to Dynamic • Yes, for Oracle databases
Possible values	Lists and describes the possible values that you can select for the property.
Default	The predefined value that is set by the external service wizard. When the property is required, you must either accept the default value or specify one yourself. If a property has no default value, the table will state No default value. The word `None` is an acceptable default value, and does not mean that there is no default value.
Unit of measure	Specifies how the property is measured, for example in kilobytes or seconds.
Property type	Describes the property type. Valid property types include: • Boolean • String • Integer
Usage	Describes usage conditions or restrictions that might apply to the property. For instance, here is how a restriction would be documented: For Rational® Application Developer for WebSphere Software version 6.40 or earlier, the password: • Must be uppercase • Must be 8 characters in length For versions of Rational Application Developer for WebSphere Software later than 6.40, the password: • Is not case sensitive • Can be up to 40 characters in length. This section lists other properties that affect this property or the properties that are affected by this property and describes the nature of the conditional relationship.

Row	Explanation
Example	Provides sample property values, for example: "If Language is set to JA (Japanese), code page number is set to `8000`".
Globalized	If a property is globalized, it has national language support, meaning that you can set the value in your national language. Valid values are **Yes** and **No**.
Bidi supported	Indicates whether the property is supported in bidirectional (bidi) processing. Bidirectional processing refers to the task of processing data that contains both right-to-left (Hebrew or Arabic, for example) and left-to-right (a URL or file path, for example) semantic content within the same file. Valid values are **Yes** and **No**.

Adapter type properties

Adapter type properties provide the external service wizard with the adapter details. These properties are configured using the external service wizard before deployment or with the WebSphere Application Server administrative console after deployment.

Note: If you set any of these adapter type properties using bidirectional script, you must set values that identify the format of the bidirectional script entered for that property.

The adapter type properties and their purpose are described in the following table. A complete description of each property is provided in the sections that follow the table. For information about how to read the property details tables in the sections that follow, see“Guide to information about properties” on page 150.

Table 10. Adapter type properties

Property name		Description
In the wizard	In the administrative console	
“Description property (Description)” on page 153	`Description`	Adapter description.
“Display Name property (DisplayName)” on page 153	`DisplayName`	Adapter display name.
“ID property (ID)” on page 153	`ID`	ID for the adapter type.
“Vendor property (Vendor)” on page 153	`Vendor`	Name of the vendor providing the adapter.
“Version property (Version)” on page 153	`Version`	Adapter version.

Description property (Description)

Adapter description.

Table 11. Description property characteristics

Required	Yes
Default	`IBM WebSphere Adapter for FTP`
Property type	String

Display Name property (DisplayName)

Adapter display name.

Table 12. DisplayName property characteristics

Required	Yes
Default	`IBM WebSphere Adapter for FTP`
Property type	String

ID property (ID)

ID for the adapter type.

Table 13. ID property characteristics

Required	Yes
Default	`FTP`
Property type	String

Vendor property (Vendor)

Name of the vendor providing the adapter.

Table 14. Vendor property characteristics

Required	Yes
Default	`IBM`
Property type	String

Version property (Version)

Adapter version.

Table 15. Version property characteristics

Required	Yes
Default	`6.2`
Property type	String

Resource adapter properties

The resource adapter properties control the general operation of the adapter, such as specifying the namespace for business objects. You set the resource adapter

properties using the external service wizard when you configure the adapter. After deploying the adapter, use the administrative console to change these properties.

The following properties for logging and tracing are no longer required in version 6.2.x, but are supported for compatibility with previous versions:

- LogFileMaxSize
- LogFileName
- LogNumberOfFiles
- TraceFileMaxSize
- TraceFileName
- TraceNumberOfFiles

The following table lists the resource adapter properties and their purpose. A complete description of each property is provided in the sections that follow the table. For information about how to read the property details tables in the sections that follow, see Guide to understanding property details.

Table 16. Resource adapter properties for the Adapter for FTP

Property name		
In the wizard	**In the administrative console**	**Description**
Adapter ID	AdapterID	Identifies the adapter instance for PMI events and for logging and tracing.
"EISEncoding (EISEncoding)" on page 155	`EISEncoding`	Encoding of the FTP server.
(Not available)	enableHASupport	Do not change this property.
"Disguise user data as "XXX" in log and trace files (HideConfidentialTrace) " on page 156	HideConfidentialTrace	Specifies whether to disguise potentially sensitive information by writing `X` strings instead of user data in the log and trace files.
(Not available)	LogFileSize	Deprecated
(Not available)	LogFilename	Deprecated
(Not available)	LogNumberOfFiles	Deprecated
(Not available)	TraceFileSize	Deprecated
(Not available)	TraceFileName	Deprecated
(Not available)	TraceNumberOfFiles	Deprecated

Adapter ID (AdapterID)

This property identifies a specific deployment or instance of the adapter.

Table 17. Adapter ID details

Required	Yes
Default	`001`
Property type	String

Table 17. Adapter ID details (continued)

Usage	This property identifies the adapter instance in the log and trace files, and also helps identify the adapter instance while monitoring adapters. The adapter ID is used with an adapter-specific identifier, `FTPRA`, to form the component name used by the Log and Trace Analyzer tool. For example, if the adapter ID property is set to `001`, the component ID is `FTPRA001`. If you run multiple instances of the same adapter, ensure that the first eight characters of the adapter ID property are unique for each instance so that you can correlate the log and trace information to a particular adapter instance. By making the first seven characters of an adapter ID property unique, the component ID for multiple instances of that adapter is also unique, allowing you to correlate the log and trace information to a particular instance of an adapter. For example, when you set the adapter ID property of two instances of WebSphere Adapter for FTP to `001` and `002`. The component IDs for those instances, `FTPRA001` and `FTPRA002`, are short enough to remain unique, enabling you to distinguish them as separate adapter instances. However, instances with longer adapter ID properties cannot be distinguished from each other. If you set the adapter ID properties of two instances to `Instance01` and `Instance02`, you will not be able to examine the log and trace information for each adapter instance because the component ID for both instances is truncated to `FTPRAInstance`. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, the value can be set both at the resource adapter level and the managed connection factory level. After you use the external service wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently. If you use the WebSphere Integration Developer assembly editor or the administrative console to reset these properties, ensure that you set them consistently, to prevent inconsistent marking of the log and trace entries.
Globalized	Yes
Bidi supported	No

EISEncoding (EISEncoding)

This property specifies the encoding of the FTP server. Sets the encoding for the control connection while communicating with the FTP server. Set the property if the FTP server's directories or file names contain globalized characters.

Table 18. EISEncoding characteristics

Required	No
Default	`None`
Property type	String
Examples	`UTF-8, ISO-8859-1`

Enable high availability support (enableHASupport)

Do not change this property. It must be set to `true`.

Disguise user data as "XXX" in log and trace files (HideConfidentialTrace)

This property specifies whether to replace user data in log and trace files with a string of `X`'s to prevent unauthorized disclosure of potentially sensitive data.

Table 19. Disguise user data as "XXX" in log and trace files details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	If you set this property to `True`, the adapter replaces user data with a string of `X`'s when writing to log and trace files. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, the value can be set both at the resource adapter level and the managed connection factory level. After you use the external service wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently. If you use the WebSphere Integration Developer assembly editor or the administrative console to reset these properties, ensure that you set them consistently, to prevent inconsistent marking of the log and trace entries.
Globalized	No
Bidi supported	No

Log file maximum size (LogFileMaxSize)

This property specifies the size of the log files in kilobytes.

Table 20. Log file maximum size details

Required	No
Default	`0`
Property type	Integer
Usage	When the log file reaches it maximum size, the adapter start using a new log file. If the file size is specified as `0` or no maximum size is specified, the file does not have a maximum size.
Globalized	Yes
Bidi supported	No

Log file name (LogFilename)

This property specifies the full path name of the log file.

Table 21. Log file name details

Required	No

Table 21. Log file name details (continued)

Default	No default value
Property type	String
Usage	This property is deprecated.
Globalized	Yes
Bidi supported	Yes

Log number of files (LogNumberOfFiles)

This property specifies the number of log files.

Table 22. Log number of files details

Required	No
Default	1
Property type	Integer
Usage	When a log file reaches its maximum size, the adapter starts using another log file. If no value is specified, the adapter creates a single log file.
Globalized	Yes
Bidi supported	No

Trace file maximum size (TraceFileMaxSize)

This property specifies the size of the trace files in kilobytes.

Table 23. Trace file maximum size details

Required	No
Default	0
Property type	Integer
Usage	If no value is specified, then the trace file has no maximum size.
Globalized	Yes
Bidi supported	No

Trace file name (TraceFilename)

This property specifies the full path of the trace file.

Table 24. Trace file name details

Required	No
Default	No default value
Unit of measure	Kilobytes
Property type	String
Usage	This property is deprecated.
Globalized	Yes
Bidi supported	Yes

Trace number of files (TraceNumberOfFiles)

This property specifies the number of trace files to use. When a trace file reaches its maximum size, the adapter starts using another trace file.

Table 25. Trace number of files details

Required	No
Default	1
Property type	Integer
Usage	If no value is specified, the adapter uses a single trace file.
Globalized	Yes
Bidi supported	No

Managed (J2C) connection factory properties

Managed connection factory properties are used by the adapter at run time to create an outbound connection instance with the FTP server.

You can set the managed connection factory properties using the external service wizard and can change them by using the WebSphere Integration Developer Assembly Editor, or after deployment through the WebSphere Process Server administrative console.

The following table lists the managed connection factory properties. A complete description of each property is provided in the sections that follow the table. For information about how to read the property details tables in the sections that follow, see Guide to understanding property details.

Note: The external service wizard refers to these properties as managed connection factory properties and the WebSphere Process Server administrative console refers to them as (J2C) connection factory properties.

Table 26. Managed connection factory properties

Property name		
In the wizard	In the administrative console	Description
Adapter ID	AdapterID	Identifies the adapter instance for PMI events and for logging and tracing.
"Custom parser class name property (CustomParserClassName)" on page 161	CustomParserClassName	Specifies the fully qualified class name of the custom parser that is used to parse the `ls –l` output.
"Data channel protection level (dataProtectionLevel)" on page 162	dataProtectionLevel	Specifies the protection level of a data channel in case of FTPS protocol.
"Default target file name property (Filename)" on page 162	Filename	Specifies the name of the file to be used during outbound operations.
"Maximum retries on connection failure (connectionRetryLimit)" on page 166	connectionRetryLimit	Specifies the number of times the adapter attempts to connect to the FTP server to reestablish the connection.

Table 26. Managed connection factory properties (continued)

"Directory property (OutputDirectory)" on page 162	OutputDirectory	Specifies the output directory in the FTP server.
"Disguise user data as "XXX" in log and trace files (HideConfidentialTrace) " on page 163	HideConfidentialTrace	Specifies whether to disguise potentially sensitive information by writing `X` strings instead of user data in the log and trace files.
Enable server verification	EnableServerVerification	Enables the remote server verification for SFTP protocol
"Encoding used by FTP server property (EISEncoding)" on page 163	EISEncoding	Specifies the encoding of the FTP server.
"FTPS connection mode property (ftpsConnectionMode)" on page 165	`ftpsConnectionMode`	Specifies the FTPS connection mode used to set up connection to the FTPS server.
Host key file	HostKeyFile	The absolute path of the host key file that contains the host keys of the trusted servers
"Host name property (HostName)" on page 166	HostName	Specifies the host name of the FTP server.
"Host name property (SecondServerHostName)" on page 167	secondServerHostName	Specifies the host name of the second FTP server.
"Host name property (SocksProxyHost)" on page 167	SocksProxyHost	Specifies the name of the workstation that is used as a proxy server.
"Keystore file property (keyStorePath)" on page 168	keyStorePath	Specifies the path of the keystore that contains the private key entries.
"Keystore password property (keyStorePassword)" on page 168	keyStorePassword	Specifies the password that is used to encrypt the keystore.
"Key password property (keyPassword)" on page 168	keyPassword	Specifies the password that is used to encrypt the key.
"Keystore type property (keyStoreType)" on page 169	keyStoreType	Specifies the type of the keystore.
Passphrase property	passPhrase	Used for enhanced security by encrypting the private key
"Password property (Password)" on page 170	Password	Specifies the password of the user with privileges to connect to the FTP server and perform FTP operations.
"Password property (SecondServerPassword)" on page 170	SecondServerPassword	Specifies the password of the Second FTP server to which the file is transferred during a server to server file transfer outbound operation.
"Password property (SocksProxyPassword)" on page 170	SocksProxyPassword	Specifies the password used to authenticate the proxy server.
"Port number property (PortNumber)" on page 171	PortNumber	Specifies the port number of the FTP server.
"Port number property (SecondServerPortNumber)" on page 171	SecondServerPortNumber	Specifies the port number of the second FTP server.

Table 26. Managed connection factory properties (continued)

"Port number property (SocksProxyPort)" on page 171	SocksProxyPort	Specifies the port number of the proxy server.
"Private key file property (PrivateKeyFilePath)" on page 171	PrivateKeyFilePath	Private key used to authenticate to the secure shell server.
"Protocol property (Protocol)" on page 172	Protocol	Specifies if the connection to the FTP server is normal FTP or secure FTP.
"Protocol property (SecondServerProtocol)" on page 172	SecondServerProtocol	Specifies the protocol used to connect to the second server.
"Connection retry interval (in milliseconds) (connectionRetryInterval)" on page 172	connectionRetryInterval	Specifies the time interval between attempts to reconnect to the FTP server if the connection fails
"Second Server Directory property (SecondServerDirectory)" on page 173	SecondServerDirectory	Specifies the directory path of the second FTP server to which the ServerToServerFileTransfer outbound operation is performed.
"Sequence file property (FileSequenceLog)" on page 173	FileSequenceLog	Specifies the full path of the file where the sequence number is stored for the outbound Create process.
"Staging directory property (StagingDirectory)" on page 174	StagingDirectory	Specifies the directory that the file is first created in to.
"Truststore file property (trustStorePath)" on page 169	trustStorePath	Specifies the path of the truststore file that contains the certificates of the FTPS servers trusted by the adapter.
"Truststore password property (trustStorePassword)" on page 169	trustStorePassword	Specifies the password of the truststore.
"User name property (SecondServerUserName)" on page 174	SecondServerUserName	Specifies the user name of the second FTP server to which the file is transferred during a server to server file transfer outbound operation.
"User Name property (SocksProxyUserName)" on page 174	SocksProxyUserName	Specifies the user name used to authenticate to the proxy server.
"User name property (Username)" on page 174	Username	Specifies the name of the user.
"Enable remote verification property (enableRemoteVerification)" on page 164	`enableRemoteVerification`	Used to verify if the host system requesting the data transfer to or from the FTP server is the same host system on which the adapter is running.

Adapter ID (AdapterID)

This property identifies a specific deployment or instance of the adapter.

Table 27. Adapter ID details

Required	Yes

Table 27. Adapter ID details (continued)

Default	`001`
Property type	String
Usage	This property identifies the adapter instance in the log and trace files, and also helps identify the adapter instance while monitoring adapters. The adapter ID is used with an adapter-specific identifier, `FTPRA`, to form the component name used by the Log and Trace Analyzer tool. For example, if the adapter ID property is set to `001`, the component ID is `FTPRA001`. If you run multiple instances of the same adapter, ensure that the first eight characters of the adapter ID property are unique for each instance so that you can correlate the log and trace information to a particular adapter instance. By making the first seven characters of an adapter ID property unique, the component ID for multiple instances of that adapter is also unique, allowing you to correlate the log and trace information to a particular instance of an adapter. For example, when you set the adapter ID property of two instances of WebSphere Adapter for FTP to `001` and `002`. The component IDs for those instances, `FTPRA001` and `FTPRA002`, are short enough to remain unique, enabling you to distinguish them as separate adapter instances. However, instances with longer adapter ID properties cannot be distinguished from each other. If you set the adapter ID properties of two instances to `Instance01` and `Instance02`, you will not be able to examine the log and trace information for each adapter instance because the component ID for both instances is truncated to `FTPRAInstance`. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, the value can be set both at the resource adapter level and the managed connection factory level. After you use the external service wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently. If you use the WebSphere Integration Developer assembly editor or the administrative console to reset these properties, ensure that you set them consistently, to prevent inconsistent marking of the log and trace entries.
Globalized	Yes
Bidi supported	No

Custom parser class name property (CustomParserClassName)

Fully qualified class name of the custom parser that is used to parse the `ls -l` output. Only used when the `ls -l` output deviates from standard output.

Table 28. Custom parser class name property characteristics

Required	No
Default	None
Property type	String
Globalized	No

Data channel protection level (dataProtectionLevel)

This property specifies the protection level of the data transferred over the data channel. It specifies the type of data channel protection that the FTP adapter and the server use.

Protection Buffer Size (PBSZ) and Data Channel Protection level (PROT) commands are issued by the FTP adapter before opening a data channel to specify the protection level on the data channel. By default, the adapter issues the "PBSZ 0" command before issuing the PROT command.

Table 29. Data channel protection level property characteristics

Required	No
Possible values	`Private - Data is transferred in encrypted form` `Clear - Data is transferred as clear text`
Default	`Private - Data is transferred in encrypted form`
Property type	String
Usage	This property is used for selecting the protection level for the data channel. Following are the protection values: • Private – Indicates that the data transfer will be integrity and confidentiality protected. • Clear – Indicates that the data channel will carry the raw data of the file transfer between the FTP adapter and the server without any security.
Globalized	No
Bidi supported	No

Default target file name property (Filename)

Specifies the name of the file that is used during outbound operations.

Table 30. Default target file name property characteristics

Required	Yes
Default	`Yes`
Property type	String
Usage	Use the WebSphere Application Server environment variable to represent the file name directory. Specify the name of the environment variable within braces, preceded by a $ symbol. For example: `${FILENAME}`. See the topic on "Defining WebSphere Application Server environment variables" on page 71 in this documentation for more information.
Globalized	No

Directory property (OutputDirectory)

This is the Output directory in the FTP Server that the outbound operation is performed on.

Table 31. Directory property characteristics

Required	No
Default	None
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the output directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${OUTPUT_DIRECTORY}`. See the topic on "Defining WebSphere Application Server environment variables" on page 71 in this documentation for more information.
Globalized	Yes

Disguise user data as "XXX" in log and trace files (HideConfidentialTrace)

This property specifies whether to replace user data in log and trace files with a string of `X`'s to prevent unauthorized disclosure of potentially sensitive data.

Table 32. Disguise user data as "XXX" in log and trace files details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	If you set this property to `True`, the adapter replaces user data with a string of `X`'s when writing to log and trace files. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, the value can be set both at the resource adapter level and the managed connection factory level. After you use the external service wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently. If you use the WebSphere Integration Developer assembly editor or the administrative console to reset these properties, ensure that you set them consistently, to prevent inconsistent marking of the log and trace entries.
Globalized	No
Bidi supported	No

Encoding used by FTP server property (EISEncoding)

Encoding of the FTP server. Use this value to set the encoding for the control connection to the FTP server.

- When both EISEncoding at the adapter level and EISEncoding at the MCF level are not set (both are null), nothing is set on the control connection while communicating with the FTP server.
- When the EISEncoding at the adapter level is set and the EISEncoding at the MCF level is not set, the value at the adapter level is set on the control connection while communicating with the FTP server. This is helpful when using

multiple MCFs, as the same encoding values are used. In this case, set the value at the adapter level so that all the connections will have the same encoding values for the control connection.

- When the EISEncoding at the adapter level is not set and the EISEncoding at the MCF level is set, the value at MCF level is set on the control connection while communicating with the FTP server. Since the value is set at the MCF level, this is applicable for only that MCF.
- When both EISEncoding at the adapter level and EISEncoding at the MCF level are set, the value at the MCF level takes precedence.

Specify any Java-supported encoding set for this attribute.

Table 33. Encoding used by FTP server property characteristics

Required	No
Default	None
Property type	String
Globalized	No

Enable server verification property (EnableServerVerification)

This property is used to enable the remote server verification for SFTP protocol.

Table 34. Enable server verification property details

Required	No
Possible values	`True` `False`
Default	False
Property type	Boolean
Usage	When this property is set to: • True, server authentication is enabled • False, server authentication is disabled The adapter checks for the HostKeyFile property in the path of the file that contains the host keys of the trusted servers.
Globalized	Yes
Bidi supported	No

Enable remote verification property (enableRemoteVerification)

When a client connects to the FTP server, two kinds of connections or channels are established; a command connection (also known as control connection), and a data connection. The command connection is the one through which the FTP commands are sent (and replies to these commands received) to the server and the data connection is the channel through which the data transfer takes place between the client and the server.

This property is used to verify if the host system requesting the data transfer to or from the FTP server is the same host system on which the adapter is running.

The verification is done while establishing a data connection to perform data transfer.

Note: This property is applicable only to FTP and FTPS protocols.

Table 35. Enable Remote verification property characteristics

Required	No
Possible values	`True` `False`
Default	`True`
Property type	Boolean
Usage	This property verifies if the data connection and the control connection are from the same host system. By default, the remote verification property is set to TRUE by the FTP server. When this property is set to: • True, during run time, the adapter checks if the data connection is established with the same host as the control connection. If the data connection is established from a different host than the control connection, then an exception is thrown and the connection fails. • False, remote verification is not performed. **Note:** Disabling the remote verification leads to low security. Precaution must be taken before disabling the remote verification.
Globalized	No
Bidi supported	No

FTPS connection mode property (ftpsConnectionMode)

This property is used to specify the connection mode when establishing a connection with the FTPS server. The WebSphere Adapter for FTP now supports both Implicit and Explicit connection modes. This property is used when you select either FTP over secure sockets layer (SSL) protocol or FTP over transport layer security (TLS) protocol.

Table 36. FTPS connection mode property characteristics

Required	No
Possible values	`Explicit` `Implicit`
Default	`Explicit`
Property type	String
Usage	This property represents the mode used to connect to the FTPS server. When this property is set to: • Explicit connection mode, initially the connection is established as a normal FTP connection. To send sensitive information, such as password the adapter switches to a secure FTP connection by issuing an `AUTH` command. **Note:** The default port for Explicit connection mode is 21. • Implicit connection mode, the connection is established as a secure FTP connection. All communications between the adapter and the server continues in a secure mode. There is no exchange of clear text information between the Adapter and the server. **Note:** The default port for Implicit connection mode is 990.
Globalized	No
Bidi supported	No

Host key file property (HostKeyFile)

This property provides the absolute path of the host key file that contains the host key of the trusted servers.

Table 37. Host key file property characteristics

Required	This property has to be specified if the EnableServerVerification property is enabled.
Default	None
Property type	String
Usage	This is used by the adapter to verify the host key of the remote server with the host keys of the trusted servers specified in this file.
Globalized	Yes
Bidi supported	No

Host name property (HostName)

Host name of the FTP Server to which the connection is established during an outbound operation.

Table 38. Host name property characteristics

Required	Yes
Default	None
Property type	String
Globalized	Yes

Maximum retries on connection failure (connectionRetryLimit)

This property specifies the number of times the adapter will attempt to reestablish a connection to the FTP server, when the adapter encounters an error related to the outbound connection.

Note: If connection timeout is configured at the FTP server, the appropriate values for connectionRetryLimit and connectionRetryInterval needs to be set. The values for properties should be set so that the FTP adapter retries the outbound request automatically if any connection error occurs due to timeout.

Table 39. Maximum retries on connection failure property characteristics

Required	No
Possible values	Integers equal to and greater than zero
Default	0
Property type	Integer

Table 39. Maximum retries on connection failure property characteristics (continued)

Usage	When this property is set to: **0** • The adapter does not attempt to reconnect to the FTP server, if an error occurs during startup or while establishing a connection. • The adapter does not verify if the connection to the FTP server is valid when there is an outbound request during runtime. **>0** • The adapter attempts to reconnect to the FTP server for the specified number of times, if an error occurs during startup or while establishing a connection. • The adapter verifies if the connection to the FTP server is valid when there is an outbound request during runtime. If the connection is not valid, it is terminated and a new connection is created to process the request. If the adapter fails to establish a connection after trying for the specified number of times, a connection error is generated. If the adapter is successful in reestablishing the connection, the outbound operation is completed.
Globalized	No
Bidi supported	No

Host name property (SecondServerHostName)

Host name of the second FTP Server to which the connection is established during an outbound operation

Table 40. Host name property characteristics

Required	Yes
Default	None
Property type	String
Usage	Contains the host name or IP address of the FTP server, for example, 9.20.13.159
Globalized	Yes

Host name property (SocksProxyHost)

Host name of the workstation that is used as a proxy server through which the adapter requests are routed to the FTP server.

Table 41. Host name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Keystore file property (keyStorePath)

This property specifies the path of the keystore that contains the private key entries.

Table 42. Keystore file property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the absolute path of the keystore file on the adapter machine (on which the adapter is running). The keystore file contains the private key entry of the FTPS client. It is also accompanied by a certificate chain for the corresponding public key. The keystore data is used to authenticate the clients identity while establishing a SSL connection.
Globalized	No
Bidi supported	No

Keystore password property (keyStorePassword)

This property specifies the password that is used to encrypt the keystore.

Table 43. Keystore password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password of the keystore. It is used to check the integrity of the keystore data. If the value is not specified, integrity check will not be executed. It is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes
Bidi supported	No

Key password property (keyPassword)

This property specifies the password that is used to encrypt the key.

Table 44. Key password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password of the key that is used to recover the key from the keystore. The property is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes
Bidi supported	No

Keystore type property (keyStoreType)

This property specifies the type of keystore.

Table 45. Keystore type property characteristics

Required	No
Possible values	JKS and PKCS12
Default	JKS
Property type	String
Usage	This property specifies the type of the keystore. It is applicable only if you select FTP over SSL or FTP over TLS as the protocol. This property is also applicable for the type of the truststore.
Globalized	No
Bidi supported	No

Truststore file property (trustStorePath)

This property specifies the path of the truststore file that contains the certificates of the FTPS servers trusted by the adapter.

Table 46. Truststore file property characteristics

Required	This property is required only if you set the protocol as FTP over SSL or FTP over TLS
Default	None
Property type	String
Usage	This property specifies the absolute path of the truststore file on the adapter machine (on which the adapter is running). The truststore file contains the certificates of FTPS servers trusted by the adapter and is used to authenticate the servers identity while establishing a SSL connection.
Globalized	No
Bidi supported	No

Truststore password property (trustStorePassword)

This property specifies the password of the truststore.

Table 47. Truststore password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password for the truststore. It is used to check the integrity of the truststore data. If the value is not specified, the integrity check will not be executed. It is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes
Bidi supported	No

Passphrase property (passPhrase)

This property is used for enhanced security by encrypting the private key.

Table 48. Passphrase property property characteristics

Required	No
Default	None
Property type	String
Usage	Used for enhanced security. It protects the private key by encrypting it in a SFTP configuration.
Globalized	Yes
Bidi supported	No

Password property (Password)

Specifies, the password of the user with privileges to connect to the FTP server and perform FTP operations.

Table 49. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password property (SecondServerPassword)

Specifies the password of the Second FTP server to which the file is transferred during a server to server file transfer outbound operation.

Table 50. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password property (SocksProxyPassword)

Specifies the password used to authenticate the proxy server.

Table 51. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Port number property (PortNumber)

Specifies the port number of the FTP server through which the connection is established during an outbound operation.

Table 52. Port number property characteristics

Required	Yes
Default	`21` for FTP and FTPS in Explicit mode, `990` for FTPS in Implicit mode, and `22` for SFTP.
Property type	Integer
Globalized	No

Port number property (SecondServerPortNumber)

Specifies the port number of the second FTP server through which the connection is established during an outbound operation.

Table 53. Port number property characteristics

Required	Yes
Default	`21` for FTP, `990` for FTPS.
Property type	Integer
Globalized	No

Port number property (SocksProxyPort)

Specifies the port number of the proxy server through which the adapter requests are routed to the FTP server.

Table 54. Port number property characteristics

Required	No
Default	`1080`
Property type	Integer
Globalized	No

Private key file property (PrivateKeyFilePath)

This property enables you to browse and select the private key, which is used to authenticate to the secure shell server.

Table 55. Private key property characteristics

Required	No
Default	None
Property type	String
Usage	Absolute path of the file which contains the private key. Used to authenticate the user to the secure shell server.
Example	`c:\temp\key.ppk`
Globalized	Yes
Bidi supported	No

Protocol property (Protocol)

Specifies the protocol that determines whether the connection to be established is a normal FTP connection or a secure FTP connection.

For example:

Normal connection: `FTP`

FTP over SSL connection: `FTPS_SSL`

FTP over TLS connection: `FTPS_TLS`

FTP over SSH connection: `SFTP`

Table 56. Protocol property characteristics

Required	Yes
Default	`FTP`
Property type	String
Globalized	No

Protocol property (SecondServerProtocol)

Specifies the protocol that is used to establish a connection to the second server. The FTP protocol is used in establishing the connection.

Table 57. Protocol property characteristics

Required	Yes
Default	`FTP`
Property type	String
Globalized	No

Connection retry interval (in milliseconds) (connectionRetryInterval)

This property specifies the time interval between attempts to reconnect to the FTP server if the connection fails.

Table 58. Connection retry interval (in milliseconds) property characteristics

Required	No
Possible values	Integers equal to and greater than 0
Default	60000
Unit of measure	Milliseconds
Property type	Integer
Usage	This property is applicable only if the value of the property "Maximum retries on connection failure" is set to greater than 0. When the adapter encounters an error while establishing a connection to the FTP server, this property specifies the time interval the adapter waits between attempts to reestablish a connection.
Globalized	No

Table 58. Connection retry interval (in milliseconds) property characteristics (continued)

Bidi supported	No

Second Server Directory property (SecondServerDirectory)

Specifies the directory of the second FTP server to which the ServerToServerFileTransfer outbound operation is performed. This is the remote event directory to which the file is transferred.

Table 59. Second Server Directory property characteristics

Required	No
Default	None
Property type	String
Usage	The directory located on the FTP server and used in outbound operation represents the absolute path of the FTP directory. It does not contain any host name or URL information. For example: `/home/usr/output`.
Globalized	Yes

Sequence file property (FileSequenceLog)

Specifies the full path of the file where the sequence number will be stored for outbound Create processing.

When the FileSequenceLog property is specified, the adapter generates a unique sequence number to insert into the file name when processing the Create operation.

The sequence of numbers will continue to increment after multiple adapter restarts.

The sequence number is inserted into the file name in the following format:

```
filename.number.extension
```

For example `Customer.3.txt`

When the FileSequenceLog property is not specified or contains an invalid value, no sequence number is generated.

Table 60. Sequence file property characteristics

Required	No
Default	None
Property type	String
Usage	**Important:** Unless they are part of a cluster, it is not recommended that two adapter instances access the same sequence file, because concurrent requests result in delay while processing batch requests.
Globalized	No

Staging directory property (StagingDirectory)

During an outbound create operation, a file is first created in the staging directory before it is moved to the directory specified in the DirectoryPath property. The staging directory is also used for the Append and Overwrite operations, where the specified file is copied to StagingDirectory (if present), then appended or overwritten with content and moved back to the original specified directory. If the StagingDirectory is not present, the operation is run in the actual required directory. When you work with a staging directory you can avoid file writing conflicts, which can occur when multiple users are reading the file or while the file is being overwritten during an append and update operation.

Table 61. Staging directory property characteristics

Required	No
Default	None
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the staging directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${STAGING_DIRECTORY}`. See the topic on "Defining WebSphere Application Server environment variables" on page 71 in this documentation for more information.
Globalized	Yes

User name property (SecondServerUserName)

Specifies the user name of the second FTP server to which the file is transferred during a server to server file transfer outbound operation.

Table 62. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

User Name property (SocksProxyUserName)

Specifies the user name used to authenticate the proxy server.

Table 63. User Name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

User name property (Username)

Specifies the name of the user with privileges to connect to the FTP server and perform FTP operations. You do not need to specify a value for this attribute if the User name is included in the URL specified in the FtpUrl property.

Table 64. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Wrapper and interaction specification properties

Wrapper properties are attributes of the wrapper business object that enable an application programmer to control an operation for the business objects in a wrapper. Interaction specification properties control the interaction for an operation for the entire adapter.

The external service wizard sets the interaction specification properties when you configure the adapter. You can change some, but not all, of these properties. However, you can change some properties for outbound operations. Use the assembly editor to change these properties, which reside in the method binding of the import. You set the wrapper properties using the WebSphere integration developer test client or programmatically at run time.

The following table lists the wrapper and interaction specification properties. A complete description of each property is provided in the sections that follow the table. For information about how to read the property details tables in the sections that follow, see Guide to understanding property details.

Table 65. Interaction specification properties

Property name		Description
In the wizard	In the administrative console	
Remote archive directory for retrieve operation	ArchiveDirectoryForRetrieve	The adapter optionally archives the file to this folder before it is deleted during a Retrieve operation.
Create new file if the file does not exist	CreateFileIfNotExists	If the file does not exist on the FTP server, the adapter creates the file when this property is set to `true` during Append and Overwrite operations.
FTP server connection mode	DataConnectionMode	Data connection mode used by the FTP server during file transfers.
Delete the file after retrieve operation	DeleteOnRetrieve	The adapter deletes the file from the FTP server after it is retrieved when this property is set to `true`.
Remote directory on FTP system	DirectoryPath	Absolute path of the directory on the FTP server where the outbound operation needs to be performed.
“Data channel protection level (dataProtectionLevel)” on page 177	dataProtectionLevel	Specifies the protection level of a data channel in case of FTPS protocol.
File content encoding	FileContentEncoding	Encoding used while writing to the file.
File in local directory	FileInLocalDirectory	If set to `true` during a create operation, the file content is picked from the local directory path of the adapter workstation.
Default target file name	Filename	Name of the file in the directory provided by the DirectoryPath property.

Table 65. Interaction specification properties (continued)

File transfer type	FileTransferType	File transfer type used during outbound operations.
Generate a unique file	GenerateUniqueFile	The adapter creates a unique file name when this property is set to `true`.
Host name property	SecondServerHostName	Hostname of the second FTP server.
Delimiter between business objects in the file property	IncludeEndBODelimiter	File content is appended with this value.
Local archive directory for create operation	LocalArchiveDirForCreate	When `LocalArchivingEnabledForCreate`is set to `true` during a create operation, the file is saved to the local workstation in this directory.
Archive file in the local directory for create operation	LocalArchivingEnabledForCreate	When set to `true`, the file is saved to the local workstation during a create operation.
Local directory	LocalDirectoryPath	The file is picked from this directory.
Port number	SecondServerPortNumber	Port number of the second FTP server.
Protocol	SecondServerProtocol	Specifies the protocol used to connect to the second server.
Script File Parameters	ScriptFileParameters	The parameters required by the FTP script file.
Directory	SecondServerDirectory	Directory path of the second FTP server during a ServerToServerFileTransfer operation.
Password	SecondServerPassword	Password of the second FTP server during a ServerToServerFileTransfer operation.
User name	SecondServerUsername	Username of the second FTP server during a ServerToServerFileTransfer operation.
Specify criteria to split file content	SplitCriteria	The delimiter that separates the business objects in the event file.
Split function class name	SplittingFunctionClassName	The fully qualified class name of the class file to be used to enable file splitting.
Staging directory	StagingDirectory	The file is first created into this directory.

Remote archive directory for retrieve operation property (ArchiveDirectory ForRetrieve)

During an outbound Retrieve operation, the adapter optionally archives the file to this folder before it is deleted. The archive directory must exist.

Table 66. Remote archive directory for retrieve operation property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Create new file if the file does not exist property (CreateFileIfNotExists)

During outbound Append and Overwrite operations, if the file does not exist on the FTP server, the adapter creates the file when this property is set to `true`. If this property is `false` and file does not exist, the adapter sends an error.

Table 67. Create new file if the file does not exist property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Directory property (SecondServerDirectory)

Directory of the second FTP server to which the server to server file transfer outbound operation is performed. This is the remote event directory to which the file is transferred.

Table 68. Directory property characteristics

Required	No
Default	None
Property type	String
Usage	For interaction specification properties, the directory located on the FTP server and used in outbound operation represents the absolute path of the FTP directory. For example: `/home/usr/output`. It does not contain any host name or URL information. For wrapper business object properties, the URL of the second server to which the ServerToServerFileTransfer outbound operation is performed. For example: The syntax for specifying the FTP URL is: `ftp://[UserId:password@]FTPserver[:port]/DirectoryForSecondServer`.
Globalized	Yes

Data channel protection level (dataProtectionLevel)

This property specifies the protection level of the data transferred over the data channel. It specifies the type of data channel protection that the FTP adapter and the server use.

Protection Buffer Size (PBSZ) and Data Channel Protection level (PROT) commands are issued by the FTP adapter before opening a data channel to specify the protection level on the data channel. By default, the adapter issues the "PBSZ 0" command before issuing the PROT command.

Table 69. Data channel protection level property characteristics

Required	No
Possible values	`Private - Data is transferred in encrypted form` `Clear - Data is transferred as clear text`
Default	`Private - Data is transferred in encrypted form`
Property type	String
Usage	This property is used for selecting the protection level for the data channel. Following are the protection values: • Private – Indicates that the data transfer will be integrity and confidentiality protected. • Clear – Indicates that the data channel will carry the raw data of the file transfer between the FTP adapter and the server without any security.
Globalized	No

Table 69. Data channel protection level property characteristics (continued)

Bidi supported	No

FTP server connection mode property (DataConnectionMode)

Data connection mode used by the FTP server during file transfers. Takes either `active` or `passive`. This value is used only when a file transfer is taking place. This property is not used when performing a server to server file transfer outbound operation.

Table 70. FTP server connection mode property characteristics

Required	No
Default	`active`
Property type	String
Possible values	`active` or `passive`
Globalized	No

Delete the file after retrieve operation (DeleteOnRetrieve)

During an outbound Retrieve operation, the adapter deletes the file from the FTP server after it is retrieved when this property is set to `true`.

Table 71. Delete the file after retrieve operation property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Remote directory on FTP system property (DirectoryPath)

Absolute path of the directory on the FTP server where the outbound operation needs to be performed for all operations except ExecuteFTPScript, or the directory path on the local adapter workstation for the ExecuteFTPScript operation only. The directory must exist.

Table 72. Remote directory on FTP system property characteristics

Required	No
Default	None
Property type	String
Usage	The DirectoryPath directory has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

File content encoding property (FileContentEncoding)

Encoding used while writing to the file. If this property is not specified, the adapter tries to read without using any specific encoding. You can specify any Java supported encoding set.

Table 73. File content encoding property characteristics

Required	No
Default	None
Property type	String
Globalized	No

File in local directory property (FileInLocalDirectory)

During outbound create operations, if this property is set to `true`, the file content is not available in the business object. The file is retrieved from the local directory on the adapter workstation. During outbound retrieve operations, if this property is set to `true`, the file content is not sent to the J2EE application as part of the business object. The file is saved to the local directory of the adapter workstation.

Table 74. File in local directory property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Default target file name property (Filename)

Name of the file to be used during outbound operations.

Table 75. Default target file name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

File transfer type property (FileTransferType)

File transfer type used during outbound operations. Takes either ASCII or binary.

Table 76. File transfer type property characteristics

Required	No
Default	`binary`
Property type	String
Globalized	No

Generate a unique file (GenerateUniqueFile)

During outbound Create operation, the adapter creates a unique file name when this property is `true`. The adapter ignores any value that is set for the Filename property when this property is set to `true`.

Note: The adapter does not support both GenerateUniqueFile and StagingDirectory options at the same time.

Table 77. Generate unique file property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No
Restrictions	The FTP server must support RFC1123 to use this feature.

Host name property (SecondServerHostName)

Host name of the second FTP server to which the connection is established during an outbound operation.

Table 78. Host name property characteristics

Required	Yes
Default	None
Property type	String
Globalized	Yes

Delimiter between business objects in the file property (IncludeEndBODelimiter)

File content is appended with this value. Used during the outbound create, append, and overwrite operations.

Table 79. Include business object delimiter in the file content property characteristics

Required	No
Default	For the create and overwrite operations, no default value is set. For the append operation, the default value is `<EndBO>`. For the append operation, the following rules apply: • If the delimiter is set to `null` in the business object wrapper, no delimiter is used to separate the business objects. • If the IncludeEndBODelimiter property is not set in the business object wrapper, and the value in the interaction specification is also `null`, the default is `<EndBO>`. • If a specific delimiter value is specified in the business object wrapper, the specified value will be appended. • If both the business object wrapper and the interaction specification have specified values, the business object wrapper value takes precedence.
Property type	String
Globalized	Yes

Local archive directory for create operation property (LocalArchiveDirForCreate)

During outbound create operations, when the file content is coming as part of the business object and LocalArchivingEnabledForCreate is set to `true`, the file is saved to the local workstation in this directory.

Table 80. Local archive directory for create property characteristics

Required	No
Default	None
Property type	String
Usage	The LocalArchiveDirForCreate directory has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Archive file in the local directory for create operation property (LocalArchivingEnabledForCreate)

During outbound create operations, when the file content is coming as part of the business object from a J2EE application and this property is set to `true`, the file is saved to the local workstation in the LocalArchiveDirForCreate directory before performing the outbound operation.

Table 81. Archive file in the local directory for create operation property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Local directory property (LocalDirectoryPath)

During outbound create operations, when FileInLocalDirectory property is set to `true`, the file content is not available in the business object. Instead the file is picked from this directory. During outbound retrieve operations, when FileInLocalDirectory property is set to `true`, the file content is not sent as part of business object. The file is saved to this directory.

Table 82. Local directory property characteristics

Required	No
Default	None
Property type	String
Usage	The LocalDirectoryPath directory has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Port number property (SecondServerPortNumber)

Port number of the second FTP server through which the connection is established during an outbound operation.

Table 83. Port number property characteristics

Required	Yes
Default	`21` for FTP, `990` for FTPS
Property type	Integer

Table 83. Port number property characteristics (continued)

Globalized	No

Protocol property (SecondServerProtocol)

Protocol that is used to establish a connection to the second server. The FTP protocol is used in establishing the connection.

Table 84. Protocol property characteristics

Required	Yes
Default	FTP
Property type	String
Globalized	No

Script File Parameters property (ScriptFileParameters)

During an outbound ExecuteFTPScript operation, the parameters required by the FTP script file are set in this property. During run time, the adapter replaces the parameters with these values.

Table 85. Script File Parameters property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password property (SecondServerPassword)

Password of the second FTP server to which the file is transferred during a server to server file transfer outbound operation.

Table 86. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

User name property (SecondServerUsername)

User name of the second FTP server to which the file is transferred during a server to server file transfer outbound operation.

Table 87. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Specify criteria to split file content property (SplitCriteria)

This property accepts different values based on the value of the SplittingFunctionClassName property.

- If the SplittingFunctionClassName property specifies that files are split based on a delimiter, then SplitCriteria contains the delimiter that separates the business objects in the event file.
- If SplittingFunctionClassName is set to a value which does splitting based on size, then the SplitCriteria property contains a valid number that represents the size in bytes.
  - If the event file size is greater than this value, the adapter splits the file into chunks of this size and the chunks are posted.
  - If the event file size is less than this value, the entire event file is posted. When SplitCriteria=0, chunking is disabled.

Table 88. Specify criteria to split file content property characteristics

Required	No
Default	0
Property type	String
Globalized	Yes

Split function class name property (SplittingFunction ClassName)

Takes the fully qualified class name of the class file to be used to enable file splitting. Requires two values:

- The com.ibm.j2ca.utils.filesplit.SplitByDelimiter class that splits the event file based on delimiter.
- The com.ibm.j2ca.utils.filesplit.SplitBySize class that splits the event file based on the event file size.

The delimiter or file size is provided in the SplitCriteria property.

Table 89. Split function class name property characteristics

Required	No
Default	com.ibm.j2ca.utils.filesplit.SplitBySize
Property type	String
Globalized	No

Staging directory property (StagingDirectory)

During outbound create operations, the file will be created in this directory first. When the file creation is complete, the file is copied to the directory specified in the DirectoryPath property. This staging directory is also used for Append and Overwrite operations where the specified file is copied to the StagingDirectory, if present. The appended or overwritten content is then moved back to the original specified directory. If StagingDirectory is not specified, the operation is run in the actual required directory.

Note: The adapter does not support both StagingDirectory and GenerateUniqueFile options at the same time.

Table 90. Staging directory property characteristics

Required	No
Default	None
Property type	String
Usage	The StagingDirectory directory has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Inbound configuration properties

WebSphere Adapter for FTP has several categories of inbound connection configuration properties, which you set with the external service wizard while generating or creating objects and services. You can change the resource adapter and activation specification properties after you deploy the module using WebSphere Integration Developer or the administrative console, but connection properties for the external service wizard cannot be changed after deployment.

Guide to information about properties

The properties used to configure WebSphere Adapter for FTP are described in detail in tables included in each of the configuration properties topics, such as Resource adapter properties, Managed connection factory properties, and so on. To help you use these tables, information about each row you might see is explained here.

The following table explains the meaning of each row that might be displayed in the table for a configuration property.

Row	**Explanation**
Required	A required field (property) must have a value in order for the adapter to work. Sometimes the external service wizard provides a default value for required properties. Removing a default value from a required field on the external service wizard *will not change that default value*. When a required field contains no value at all, the external service wizard processes the field using its assigned default value, and that default value is displayed on the administrative console. Possible values are **Yes** and **No**. Sometimes a property is required only when another property has a specific value. When this is the case, the table will note this dependency. For example, • Yes, when the EventQueryType property is set to Dynamic • Yes, for Oracle databases
Possible values	Lists and describes the possible values that you can select for the property.

Row	Explanation
Default	The predefined value that is set by the external service wizard. When the property is required, you must either accept the default value or specify one yourself. If a property has no default value, the table will state No default value. The word `None` is an acceptable default value, and does not mean that there is no default value.
Unit of measure	Specifies how the property is measured, for example in kilobytes or seconds.
Property type	Describes the property type. Valid property types include: • Boolean • String • Integer
Usage	Describes usage conditions or restrictions that might apply to the property. For instance, here is how a restriction would be documented: For Rational Application Developer for WebSphere Software version 6.40 or earlier, the password: • Must be uppercase • Must be 8 characters in length For versions of Rational Application Developer for WebSphere Software later than 6.40, the password: • Is not case sensitive • Can be up to 40 characters in length. This section lists other properties that affect this property or the properties that are affected by this property and describes the nature of the conditional relationship.
Example	Provides sample property values, for example: "If Language is set to JA (Japanese), code page number is set to `8000`".
Globalized	If a property is globalized, it has national language support, meaning that you can set the value in your national language. Valid values are **Yes** and **No**.
Bidi supported	Indicates whether the property is supported in bidirectional (bidi) processing. Bidirectional processing refers to the task of processing data that contains both right-to-left (Hebrew or Arabic, for example) and left-to-right (a URL or file path, for example) semantic content within the same file. Valid values are **Yes** and **No**.

Adapter type properties

Adapter type properties provide the external service wizard with the adapter details. These properties are configured using the external service wizard before deployment or with the WebSphere Application Server administrative console after deployment.

Note: If you set any of these adapter type properties using bidirectional script, you must set values that identify the format of the bidirectional script entered for that property.

The adapter type properties and their purpose are described in the following table. A complete description of each property is provided in the sections that follow the table. For information about how to read the property details tables in the sections that follow, see"Guide to information about properties" on page 150.

Table 91. Adapter type properties

Property name		
In the wizard	In the administrative console	Description
"Description property (Description)"	`Description`	Adapter description.
"Display Name property (DisplayName)"	`DisplayName`	Adapter display name.
"ID property (ID)"	`ID`	ID for the adapter type.
"Vendor property (Vendor)" on page 187	`Vendor`	Name of the vendor providing the adapter.
"Version property (Version)" on page 187	`Version`	Adapter version.

Description property (Description)

Adapter description.

Table 92. Description property characteristics

Required	Yes
Default	`IBM WebSphere Adapter for FTP`
Property type	String

Display Name property (DisplayName)

Adapter display name.

Table 93. DisplayName property characteristics

Required	Yes
Default	`IBM WebSphere Adapter for FTP`
Property type	String

ID property (ID)

ID for the adapter type.

Table 94. ID property characteristics

Required	Yes

Table 94. ID property characteristics (continued)

Default	FTP
Property type	String

Vendor property (Vendor)

Name of the vendor providing the adapter.

Table 95. Vendor property characteristics

Required	Yes
Default	IBM
Property type	String

Version property (Version)

Adapter version.

Table 96. Version property characteristics

Required	Yes
Default	6.2
Property type	String

Resource adapter properties

The resource adapter properties control the general operation of the adapter, such as specifying the namespace for business objects. You set the resource adapter properties using the external service wizard when you configure the adapter. After deploying the adapter, use the administrative console to change these properties.

The following properties for logging and tracing are no longer required in version 6.2.x, but are supported for compatibility with previous versions:

- LogFileMaxSize
- LogFileName
- LogNumberOfFiles
- TraceFileMaxSize
- TraceFileName
- TraceNumberOfFiles

The following table lists the resource adapter properties and their purpose. A complete description of each property is provided in the sections that follow the table. For information about how to read the property details tables in the sections that follow, see Guide to understanding property details.

Table 97. Resource adapter properties for the Adapter for FTP

Property name		
In the wizard	**In the administrative console**	**Description**
Adapter ID	AdapterID	Identifies the adapter instance for PMI events and for logging and tracing.

Table 97. Resource adapter properties for the Adapter for FTP (continued)

Property name		
In the wizard	**In the administrative console**	**Description**
"EISEncoding (EISEncoding)" on page 189	EISEncoding	Encoding of the FTP server.
(Not available)	enableHASupport	Do not change this property.
"Disguise user data as "XXX" in log and trace files (HideConfidentialTrace) " on page 190	HideConfidentialTrace	Specifies whether to disguise potentially sensitive information by writing X strings instead of user data in the log and trace files.
(Not available)	LogFileSize	Deprecated
(Not available)	LogFilename	Deprecated
(Not available)	LogNumberOfFiles	Deprecated
(Not available)	TraceFileSize	Deprecated
(Not available)	TraceFileName	Deprecated
(Not available)	TraceNumberOfFiles	Deprecated

Adapter ID (AdapterID)

This property identifies a specific deployment or instance of the adapter.

Table 98. Adapter ID details

Required	Yes
Default	001
Property type	String

Table 98. Adapter ID details (continued)

Usage	This property identifies the adapter instance in the log and trace files, and also helps identify the adapter instance while monitoring adapters. The adapter ID is used with an adapter-specific identifier, `FTPRA`, to form the component name used by the Log and Trace Analyzer tool. For example, if the adapter ID property is set to `001`, the component ID is `FTPRA001`. If you run multiple instances of the same adapter, ensure that the first eight characters of the adapter ID property are unique for each instance so that you can correlate the log and trace information to a particular adapter instance. By making the first seven characters of an adapter ID property unique, the component ID for multiple instances of that adapter is also unique, allowing you to correlate the log and trace information to a particular instance of an adapter. For example, when you set the adapter ID property of two instances of WebSphere Adapter for FTP to `001` and `002`. The component IDs for those instances, `FTPRA001` and `FTPRA002`, are short enough to remain unique, enabling you to distinguish them as separate adapter instances. However, instances with longer adapter ID properties cannot be distinguished from each other. If you set the adapter ID properties of two instances to `Instance01` and `Instance02`, you will not be able to examine the log and trace information for each adapter instance because the component ID for both instances is truncated to `FTPRAInstance`. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, the value can be set both at the resource adapter level and the managed connection factory level. After you use the external service wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently. If you use the WebSphere Integration Developer assembly editor or the administrative console to reset these properties, ensure that you set them consistently, to prevent inconsistent marking of the log and trace entries.
Globalized	Yes
Bidi supported	No

EISEncoding (EISEncoding)

This property specifies the encoding of the FTP server. Sets the encoding for the control connection while communicating with the FTP server. Set the property if the FTP server's directories or file names contain globalized characters.

Table 99. EISEncoding characteristics

Required	No
Default	`None`
Property type	String
Examples	`UTF-8, ISO-8859-1`

Enable high availability support (enableHASupport)

Do not change this property. It must be set to `true`.

Disguise user data as "XXX" in log and trace files (HideConfidentialTrace)

This property specifies whether to replace user data in log and trace files with a string of `X`'s to prevent unauthorized disclosure of potentially sensitive data.

Table 100. Disguise user data as "XXX" in log and trace files details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	If you set this property to `True`, the adapter replaces user data with a string of `X`'s when writing to log and trace files. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, the value can be set both at the resource adapter level and the managed connection factory level. After you use the external service wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently. If you use the WebSphere Integration Developer assembly editor or the administrative console to reset these properties, ensure that you set them consistently, to prevent inconsistent marking of the log and trace entries.
Globalized	No
Bidi supported	No

Log file maximum size (LogFileMaxSize)

This property specifies the size of the log files in kilobytes.

Table 101. Log file maximum size details

Required	No
Default	`0`
Property type	Integer
Usage	When the log file reaches it maximum size, the adapter start using a new log file. If the file size is specified as `0` or no maximum size is specified, the file does not have a maximum size.
Globalized	Yes
Bidi supported	No

Log file name (LogFilename)

This property specifies the full path name of the log file.

Table 102. Log file name details

Required	No

Table 102. Log file name details (continued)

Default	No default value
Property type	String
Usage	This property is deprecated.
Globalized	Yes
Bidi supported	Yes

Log number of files (LogNumberOfFiles)

This property specifies the number of log files.

Table 103. Log number of files details

Required	No
Default	1
Property type	Integer
Usage	When a log file reaches its maximum size, the adapter starts using another log file. If no value is specified, the adapter creates a single log file.
Globalized	Yes
Bidi supported	No

Trace file maximum size (TraceFileMaxSize)

This property specifies the size of the trace files in kilobytes.

Table 104. Trace file maximum size details

Required	No
Default	0
Property type	Integer
Usage	If no value is specified, then the trace file has no maximum size.
Globalized	Yes
Bidi supported	No

Trace file name (TraceFilename)

This property specifies the full path of the trace file.

Table 105. Trace file name details

Required	No
Default	No default value
Unit of measure	Kilobytes
Property type	String
Usage	This property is deprecated.
Globalized	Yes
Bidi supported	Yes

Trace number of files (TraceNumberOfFiles)

This property specifies the number of trace files to use. When a trace file reaches its maximum size, the adapter starts using another trace file.

Table 106. Trace number of files details

Required	No
Default	1
Property type	Integer
Usage	If no value is specified, the adapter uses a single trace file.
Globalized	Yes
Bidi supported	No

Activation specification properties

Activation specification properties are properties that hold the inbound event processing configuration information for a message endpoint.

Activation specification properties are used during endpoint activation to notify the adapter of eligible event listeners. During inbound processing, the adapter uses these event listeners to receive events before forwarding them to the endpoint (a message driven bean).

You set the activation specification properties using the external service wizard and can change them using the WebSphere Integration Developer Assembly Editor, or after deployment through the administrative console.

The following table lists the activation specification properties. A complete description of each property is provided in the sections that follow the table. For information about how to read the property details tables in the sections that follow, see Guide to understanding property details.

Table 107. Activation specification properties

Property name		
In the wizard	In the administrative console	Description
"Ensure once-only event delivery (AssuredOnceDelivery)" on page 196	AssuredOnceDelivery	Specifies whether the adapter provides assured once delivery of events.
"Auto create event table property (EP_CreateTable)" on page 197	`EP_CreateTable`	Tells the adapter whether to create the Event Persistence table
"Create Table property (CreateTable)" on page 197	`CreateTable`	When set to `true`, the event table and related indexes are created
"Custom parser class name property (CustomParserClassName)" on page 197	`CustomParserClassName`	Fully qualified class name of the custom parser which is used to parse the `ls –l` output
"Data channel protection level (dataProtectionLevel)" on page 197	dataProtectionLevel	Specifies the protection level of a data channel in case of FTPS protocol.
"Database Password property (DatabasePassword)" on page 198	`DatabasePassword`	Password used by event persistence for retrieving the JDBC database connection from the data source

Table 107. Activation specification properties (continued)

"Database schema name property (EP_SchemaName)" on page 198	`EP_SchemaName`	Schema name of the database used by event persistence
"Database Username property (DatabaseUsername)" on page 198	`DatabaseUsername`	User name used by event persistence for retrieving the JDBC database connection from the data source
"FTP server connection mode property (DataConnectionMode)" on page 200	`DataConnectionMode`	Data connection mode used by the FTP server during file transfers
"FTPS connection mode property (ftpsConnectionMode)" on page 200	`ftpsConnectionMode`	Specifies the FTPS connection mode used to set up connection to the FTPS server.
(Not available)	`DefaultObjectName`	Supported for compatibility with earlier versions
"Delivery type (DeliveryType)" on page 198	DeliveryType	Determines the order in which events are delivered by the adapter to the export.
"Encoding used by FTP server property (EISEncoding)" on page 199	`EISEncoding`	Encoding of the FTP server
(Not available)	`EventContentType`	Supported for compatibility with earlier versions
"Event recovery data source (JNDI) name property (EP_DataSource_JNDIName)" on page 199	`EP_DataSource_JNDIName`	JNDI name of the data source used by event persistence to get the JDBC database connection
"Event recovery table name property (EP_EventTableName)" on page 200	`EP_TableName`	Name of the table that is used by the adapter for event persistence
"Failure file extension for local archive property (FailedArchiveExt)" on page 201	`FailedArchiveExt`	File extension used to archive business objects in the event file that are not successfully processed
"File content encoding property (FileContentEncoding)" on page 201	`FileContentEncoding`	Encoding used to read the event files
"File extension for remote archive property (ftpRenameExt)" on page 201	`ftpRenameExt`	File extension or suffix that the adapter uses to rename the remote FTP file
"Keystore file property (keyStorePath)" on page 202	keyStorePath	Specifies the path of the keystore that contains the private key entries.
"Keystore password property (keyStorePassword)" on page 202	keyStorePassword	Specifies the password that is used to encrypt the keystore.
"Key password property (keyPassword)" on page 202	keyPassword	Specifies the password that is used to encrypt the key.
"Keystore type property (keyStoreType)" on page 203	keyStoreType	Specifies the type of the keystore.
"Pass only file name and directory, not the content property (FilePassByReference)" on page 204	`FilePassByReference`	Specifies that the file content of the event file is not sent to the export
"File transfer type property (FileTransferType)" on page 204	`FileTransferType`	File transfer type used during inbound processing
"Number of files to get at a time property (ftpGetQuantity)" on page 204	`ftpGetQuantity`	Determines the number of files retrieved from the remote FTP URL

Table 107. Activation specification properties (continued)

"Number of poll periods between downloads property (ftpPollFrequency)" on page 204	`ftpPollFrequency`	Determines how frequently the adapter polls the FTP server
Retry limit for failed events	FailedEventRetryLimit	The number of times the adapter attempts to redeliver an event before marking the event as failed.
"Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)" on page 205	`ftpScriptFileExecutedAfterInbound`	Specifies the path of the script file that will be executed after downloading the files from the FTP server
"Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)" on page 206	`ftpScriptFileExecutedBeforeInbound`	Specifies the path of the script file that will be executed before downloading the files from the FTP server
"Host name property (HostName)" on page 206	`HostName`	Host name of the FTP Server to which the connection is established
"Include business object delimiter in the file content property (IncludeEndBODelimiter)" on page 206	`IncludeEndBODelimiter`	When set to `true`, the delimiter is sent with the business object content for further processing
"Local archive directory property (LocalArchiveDirectory)" on page 206	`LocalArchiveDirectory`	Absolute path of the local Archive directory
"Local directory property (LocalEventDirectory)" on page 207	`LocalEventDirectory`	Local system directory into which the adapter downloads event files from the FTP site
"Maximum connections (MaximumConnections)" on page 207	MaximumConnections	The maximum number of connections that the adapter can use for inbound event delivery.
"Minimum connections (MinimumConnections)" on page 207	MinimumConnections	The minimum number of connections that the adapter can use for inbound event delivery.
"File extension for local archive property (originalArchiveExt)" on page 208	`OriginalArchiveExt`	File extension used to archive the original event file
Passphrase property	passPhrase	Used for enhanced security by encrypting the private key
"Password property (Password)" on page 208	`Password`	Password of the user who has privileges to connect to the FTP server and perform FTP operations
"Password used to connect to event data source property (EP_Password)" on page 208	`EP_Password`	Password used during event persistence
"Interval between polling periods (PollPeriod)" on page 209	PollPeriod	The length of time that the adapter waits between polling periods.
"Maximum events in polling period (PollQuantity)" on page 209	PollQuantity	The number of events the adapter delivers to the export during each poll period.
"Port number property (PortNumber)" on page 210	`PortNumber`	Port number of the FTP server
"Private key file property (PrivateKeyFilePath)" on page 210	`PrivateKeyFilePath`	Private key used to authenticate to the Secure shell server

Table 107. Activation specification properties (continued)

"Protocol property (Protocol)" on page 210	`Protocol`	Specifies if the connection to the FTP server is normal FTP or secure FTP.
"Retrieve files with this pattern property (EventFileMask)" on page 212	`EventFileMask`	Filter for the event files
Retry EIS connection on startup	RetryConnectionOnStartup	Controls whether the adapter retries the connection to the FTP server if it cannot connect at startup.
Time between retries in case of system connection failure (milliseconds)	RetryInterval	The length of time that the adapter waits between attempts to establish a new connection after an error during inbound operations.
Maximum number of retries in case of system connection failure	RetryLimit	The number of times the adapter tries to reestablish an inbound connection after an error.
"Remote archive directory property (ftpArchiveDirectory)" on page 211	`ftpArchiveDirectory`	Relative path of the archive directory on the FTP server
"Remote directory property (EventDirectory)" on page 211	`EventDirectory`	Remote directory of the FTP server from where the event files are retrieved for inbound processing
Enable server verification	EnableServerVerification	Enables the remote server verification for SFTP protocol
Host key file	HostKeyFile	The absolute path of the host key file that contains the host keys of the trusted servers
"Host name property (SocksProxyHost)" on page 215	`SocksProxyHost`	Host name of the machine used as a proxy server
"Password property (SocksProxyPassword)" on page 215	`SocksProxyPassword`	Password used to authenticate the proxy server
"Port number property (SocksProxyPort)" on page 216	`SocksProxyPort`	Port number of the proxy server
"User name property (SocksProxyUserName)" on page 216	`SocksProxyUserName`	User name used to authenticate the proxy server
"Sort event files property (SortEventFiles)" on page 216	`SortEventFiles`	Determines the sorting order of event files being polled
"Specify criteria to split file content property (SplitCriteria)" on page 216	`SplitCriteria`	Takes different values based on the value of the SplittingFunctionClassName property
"Splitting function class name property" on page 218	`SplittingFunctionClassName`	Takes the fully qualified class name of the class file to be used to enable file splitting
"Stop the adapter when an error is encountered while polling (StopPollingOnError)" on page 218	StopPollingOnError	Specifies whether the adapter stops polling for events when it encounters an error during polling.
"Success file extension for local archive property (SuccessArchiveExt)" on page 219	`SuccessArchiveExt`	File extension used to archive all the successfully processed business objects

Table 107. Activation specification properties (continued)

"Truststore file property (trustStorePath)" on page 203	trustStorePath	Specifies the path of the truststore file that contains the certificates of the FTPS servers trusted by the adapter.
"Truststore password property (trustStorePassword)" on page 203	trustStorePassword	Specifies the password of the truststore.
"User name property (UserName)" on page 219	`UserName`	Name of the user who has privileges to connect to the FTP server and perform FTP operations
"User name used to connect to event data source property (EP_UserName)" on page 219	`EP_UserName`	User name used by event persistence for getting the database connection
Rule editor to filter files	ruleString	The collection of rules used to filter the events.
"Enable remote verification property (enableRemoteVerification)" on page 212	`enableRemoteVerification`	Used to verify if the host system requesting the data transfer to or from the FTP server is the same host system on which the adapter is running.

Ensure once-only event delivery (AssuredOnceDelivery)

This property specifies whether to provide ensure once-only event delivery for inbound events.

Table 108. Ensure once-only event delivery details

Required	Yes
Possible values	`True` `False`
Default	`True`
Property type	Boolean
Usage	When this property is set to `True`, the adapter provides assured once event delivery. This means that each event will be delivered once and only once. A value of `False` does not provide assured once event delivery, but provides better performance. When this property is set to `True`, the adapter attempts to store transaction (XID) information in the event store. If it is set to `False`, the adapter does not attempt to store the information. This property is used only if the export component is transactional. If it is not, no transaction can be used, regardless of the value of this property.
Globalized	No
Bidi supported	No

Auto create event table property (EP_CreateTable)

Tells the adapter whether to create the Event Persistence table. If the value is `true` and table does not exist then the adapter creates the table. If the value is `false` the adapter does not create the table.

Table 109. Auto create event table property characteristics

Required	No
Default	`true`
Property type	Boolean
Globalized	No

Create Table property (CreateTable)

When set to `true`, the event table and related indexes are created. For troubleshooting table creation errors, set this property to `false`. The table and indexes can then be created manually.

Table 110. Create Table property characteristics

Required	No
Default	`true`
Property type	Boolean
Globalized	No

Custom parser class name property (CustomParserClassName)

Fully qualified class name of the custom parser which is used to parse the `ls –l` output. Used only when the `ls –l` output deviates from standard output.

Table 111. Custom parser class name property characteristics

Required	No
Default	None
Property type	String
Globalized	No

Data channel protection level (dataProtectionLevel)

This property specifies the protection level of the data transferred over the data channel. It specifies the type of data channel protection that the FTP adapter and the server use.

Protection Buffer Size (PBSZ) and Data Channel Protection level (PROT) commands are issued by the FTP adapter before opening a data channel to specify the protection level on the data channel. By default, the adapter issues the "PBSZ 0" command before issuing the PROT command.

Table 112. Data channel protection level property characteristics

Required	No
Possible values	`Private - Data is transferred in encrypted form` `Clear - Data is transferred as clear text`

Table 112. Data channel protection level property characteristics (continued)

Default	`Private - Data is transferred in encrypted form`
Property type	String
Usage	This property is used for selecting the protection level for the data channel. Following are the protection values: • Private – Indicates that the data transfer will be integrity and confidentiality protected. • Clear – Indicates that the data channel will carry the raw data of the file transfer between the FTP adapter and the server without any security.
Globalized	No
Bidi supported	No

Database Password property (DatabasePassword)

Password used by event persistence for retrieving the JDBC database connection from the data source.

Table 113. Database Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Database schema name property (EP_SchemaName)

Schema name of the database used by event persistence.

Table 114. Database schema name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Database Username property (DatabaseUsername)

User name used by event persistence for retrieving the JDBC database connection from the data source.

Table 115. Database username property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Delivery type (DeliveryType)

This property specifies the order in which events are delivered by the adapter to the export.

Table 116. Delivery type details

Required	No
Possible values	`ORDERED` `UNORDERED`
Default	`ORDERED`
Property type	String
Usage	The following values are supported: • `ORDERED`: The adapter delivers events to the export one at a time. • `UNORDERED`: The adapter delivers all events to the export at once.
Globalized	No
Bidi supported	No

Encoding used by FTP server property (EISEncoding)

Encoding of the FTP server. Use this value to set the encoding for the control connection to the FTP server.

- When both EISEncoding at the adapter level and EISEncoding at the activation specification level are not set (both are null), nothing is set on the control connection while communicating with the FTP server.
- When EISEncoding at the adapter level is set and EISEncoding at the activation specification level is not set, the value at adapter level is set on the control connection while communicating with the FTP server. This is helpful when using multiple activation specifications and the same encoding is set. In this case, set the value at the adapter level so that all the connections have the same encoding for the control connection.
- When EISEncoding at the adapter level is not set and EISEncoding at the activation specification level is set, the value at activation specification level is set on the control connection while communicating with the FTP server. Since the value is at the activation specification level, this is applicable for only that activation specification.
- When both EISEncoding at the adapter level and EISEncoding at the activation specification level are set, the value at the activation specification level takes precedence.

Specify any Java-supported encoding set for this attribute.

Table 117. Encoding used by FTP server property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Event recovery data source (JNDI) name property (EP_DataSource_JNDIName)

JNDI name of the data source used by event persistence to get the JDBC database connection. The data source must be created in WebSphere Process Server. The database name specified while creating the data source must exist.

Table 118. Event recovery data source (JNDI) name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Event recovery table name property (EP_EventTableName)

Name of the table that is used by the adapter for event persistence. When using multiple activation specifications, this value must be unique for each. The same table name must not be used by other instances of same adapter or a different adapter. If the table does not exist in the database, the adapter will create the table.

Table 119. Event recovery table name property characteristics

Required	No
Default	`FTPTABLE`
Property type	String
Globalized	Yes

FTP server connection mode property (DataConnectionMode)

Data connection mode used by the FTP server during file transfers. Accepts either active or passive settings.

Table 120. FTP server connection mode property characteristics

Required	No
Default	`active`
Property type	String
Globalized	No

FTPS connection mode property (ftpsConnectionMode)

This property is used to specify the connection mode when establishing a connection with the FTPS server. The WebSphere Adapter for FTP now supports both Implicit and Explicit connection modes. This property is used when you select either FTP over secure sockets layer (SSL) protocol or FTP over transport layer security (TLS) protocol.

Table 121. FTPS connection mode property characteristics

Required	No
Possible values	`Explicit` `Implicit`
Default	`Explicit`
Property type	String

Table 121. FTPS connection mode property characteristics (continued)

Usage	This property represents the mode used to connect to the FTPS server. When this property is set to: • Explicit connection mode, initially the connection is established as a normal FTP connection. To send sensitive information, such as password the adapter switches to a secure FTP connection by issuing an `AUTH` command. **Note:** The default port for Explicit connection mode is 21. • Implicit connection mode, the connection is established as a secure FTP connection. All communications between the adapter and the server continues in a secure mode. There is no exchange of clear text information between the Adapter and the server. **Note:** The default port for Implicit connection mode is 990.
Globalized	No
Bidi supported	No

Failure file extension for local archive property (FailedArchiveExt)

File extension used to archive business objects in the event file that are not successfully processed. This property is used only when LocalArchiveDirectory is valid and exists.

Table 122. Failure file extension for local archive property characteristics

Required	No
Default	`fail`
Property type	String
Globalized	Yes

File content encoding property (FileContentEncoding)

Encoding used to read the event files based on the EndBODelimiter property and during string to byte[] conversions. If not specified, the adapter attempts to read without any specific encoding. You can specify any Java supported encoding set.

Table 123. File content encoding property characteristics

Required	No
Default	None
Property type	String
Globalized	No

File extension for remote archive property (ftpRenameExt)

File extension or suffix that the adapter uses to rename the remote FTP file after the connector has polled for it. Renaming the file prevents the connector from polling the same file in the next poll cycle. The adapter can be configured to rename the processed event file and move it to an archive directory.

Table 124. File extension for remote archive property characteristics

Required	No
Default	None

Table 124. File extension for remote archive property characteristics (continued)

Property type	String
Globalized	Yes

Keystore file property (keyStorePath)

This property specifies the path of the keystore that contains the private key entries.

Table 125. Keystore file property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the absolute path of the keystore file on the adapter machine (on which the adapter is running). The keystore file contains the private key entry of the FTPS client. It is also accompanied by a certificate chain for the corresponding public key. The keystore data is used to authenticate the clients identity while establishing a SSL connection.
Globalized	No
Bidi supported	No

Keystore password property (keyStorePassword)

This property specifies the password that is used to encrypt the keystore.

Table 126. Keystore password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password of the keystore. It is used to check the integrity of the keystore data. If the value is not specified, integrity check will not be executed. It is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes
Bidi supported	No

Key password property (keyPassword)

This property specifies the password that is used to encrypt the key.

Table 127. Key password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password of the key that is used to recover the key from the keystore. The property is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes

Table 127. Key password property characteristics (continued)

Bidi supported	No

Keystore type property (keyStoreType)

This property specifies the type of keystore.

Table 128. Keystore type property characteristics

Required	No
Possible values	JKS and PKCS12
Default	JKS
Property type	String
Usage	This property specifies the type of the keystore. It is applicable only if you select FTP over SSL or FTP over TLS as the protocol. This property is also applicable for the type of the truststore.
Globalized	No
Bidi supported	No

Truststore file property (trustStorePath)

This property specifies the path of the truststore file that contains the certificates of the FTPS servers trusted by the adapter.

Table 129. Truststore file property characteristics

Required	This property is required only if you set the protocol as FTP over SSL or FTP over TLS
Default	None
Property type	String
Usage	This property specifies the absolute path of the truststore file on the adapter machine (on which the adapter is running). The truststore file contains the certificates of FTPS servers trusted by the adapter and is used to authenticate the servers identity while establishing a SSL connection.
Globalized	No
Bidi supported	No

Truststore password property (trustStorePassword)

This property specifies the password of the truststore.

Table 130. Truststore password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password for the truststore. It is used to check the integrity of the truststore data. If the value is not specified, the integrity check will not be executed. It is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes

Table 130. Truststore password property characteristics (continued)

Bidi supported	No

Pass only file name and directory, not the content property (FilePassByReference)

Specifies that the file content of the event file is not sent to the export.

If set to `true`, the file is appended with a timestamp and sent to the LocalArchiveDirectory. The timestamp prevents errors and overwrites to the file when another file with the same name is received. This property can be set to `true` only when the LocalArchiveDirectory property is set and the specified directory exists. This property is used only for PassThrough inbound processing. When enabled, the file is not split into chunks.

Table 131. Pass only file name and directory, not the content property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

File transfer type property (FileTransferType)

File transfer type used during inbound processing. Accepts either ASCII or binary.

Table 132. File transfer type property characteristics

Required	No
Default	`binary`
Property type	String
Globalized	no

Number of files to get at a time property (ftpGetQuantity)

Determines the number of files retrieved from the remote FTP URL with each remote poll.

Table 133. Number of files to get at a time property characteristic

Required	Yes
Default	`10`
Property type	Integer
Globalized	No

Number of poll periods between downloads property (ftpPollFrequency)

Determines how frequently the adapter polls the FTP server, measured in the number of standard poll cycles. For example, if PollPeriod is set to 10000, and FTPPollFrequency is set to 6, the adapter polls the LocalEventDirectory every 10 seconds and polls the remote EventDirectory every 60 seconds. The adapter

performs FTP polling only if you specify a value for this property. If PollPeriod is 0, you consider it as 1 for calculation. If the calculation evaluates to 0, the adapter does not perform FTP polling.

Table 134. Number of poll periods between downloads property characteristics

Required	Yes
Default	5
Property type	Integer
Globalized	No

Retry limit for failed events (FailedEventRetryLimit)

This property specifies the number of times that the adapter attempts to redeliver an event before marking the event as failed.

Table 135. Retry limit for failed events details

Required	No
Possible values	Integers
Default	5
Property type	Integer
Usage	Use this property to control how many times the adapter tries to send an event before marking it as failed. It accepts the following values: **Default** If this property is not set, the adapter tries five additional times before marking the event as failed. **0** The adapter tries to deliver the event an infinite number of times. When the property is set to 0, the event remains in the event store and the event is never marked as failed. **> 0** For integers greater than zero, the adapter retries the specified number of times before marking the event as failed. **< 0** For negative integers, the adapter does not retry failed events.
Globalized	No
Bidi supported	No

Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)

Specifies the path of the script file that will be executed after downloading the files from the FTP server.

Table 136. Run FTP script file after downloading files property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)

Specifies the path of the script file that will be executed before downloading the files from the FTP server.

Table 137. Run FTP script file before downloading files property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Host name property (HostName)

Host name of the FTP Server to which the connection is established during inbound processing.

Table 138. Create Table property characteristics

Required	Yes
Default	None
Property type	String
Globalized	Yes

Include business object delimiter in the file content property (IncludeEndBODelimiter)

When set to `true`, the delimiter is sent with the business object content for further processing. This property is valid only when splitting the event files based on a delimiter.

Table 139. Include business object delimiter in the file content property characteristics

Required	No
Default	`false`
Property type	String
Globalized	No

Local archive directory property (LocalArchiveDirectory)

Absolute path of the local Archive directory. The directory must be valid and exist.

Table 140. Local archive directory property characteristics

Required	No
Default	None
Property type	String

Table 140. Local archive directory property characteristics (continued)

Usage	You can use a WebSphere Application Server environment variable to represent the local archive directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${LOCALARCHIVE_DIRECTORY}`. See the topic on "Defining WebSphere Application Server environment variables" on page 71 in this documentation. **Note:** The **LocalArchiveDirectory** has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Local directory property (LocalEventDirectory)

Local system directory into which the adapter downloads event files from the FTP site. You must specify a value for this property to enable the adapter to process events.

Table 141. Local directory property characteristics

Required	Yes
Default	None
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the local event directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${LOCAL_DIRECTORY}`. See the topic on "Defining WebSphere Application Server environment variables" on page 71 in this documentation. **Note:** The **LocalEventDirectory** has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Maximum connections (MaximumConnections)

This property specifies the maximum number of connections that the adapter can use for inbound event delivery.

Table 142. Maximum connections details

Required	No
Default	1
Property type	Integer
Usage	Only positive values are valid. The adapter considers any positive entry less than 1 to be equal to 1. Typing a negative value for this property may result in run time errors.
Globalized	No
Bidi supported	No

Minimum connections (MinimumConnections)

This property specifies the minimum number of connections that the adapter can use for inbound event delivery.

Table 143. Minimum connections details

Required	No
Default	1
Property type	Integer
Usage	Only positive values are valid. Any value less than 1 is treated as 1 by the adapter. Typing a negative value or 1 for this property may result in run time errors.
Globalized	No
Bidi supported	No

File extension for local archive property (originalArchiveExt)

File extension used to archive the original event file. This preserves the entire event file for reference in case any of its business objects fail. This property is used only when LocalArchiveDirectory is valid and exists.

Table 144. File extension for local archive property characteristics

Required	No
Default	`original`
Property type	String
Globalized	Yes

Password property (Password)

Password of the user who has privileges to connect to the FTP server and perform FTP operations. You do not need to specify a value for this property if the password is included in the URL specified in the EventDirectory property.

Table 145. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password used to connect to event data source property (EP_Password)

The password used during event persistence to get the database connection from the data source.

Table 146. Password used to connect to event data source property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Interval between polling periods (PollPeriod)

This property specifies the length of time that the adapter waits between polling periods.

Table 147. Interval between polling periods details

Required	Yes
Possible values	Integers greater than or equal to 0.
Default	2000
Unit of measure	Milliseconds
Property type	Integer
Usage	The poll period is established at a fixed rate, which means that if running the poll cycle is delayed for any reason (for example, if a prior poll cycle takes longer than expected to complete) the next poll cycle will occur immediately to make up for the lost time caused by the delay.
Globalized	No
Bidi supported	No

Maximum events in polling period (PollQuantity)

This property specifies the number of events that the adapter delivers to the export during each poll period.

Table 148. Maximum events in polling period details

Required	Yes
Default	10
Property type	Integer
Usage	The value must be greater than 0. If this value is increased, more events are processed per polling period and the adapter may perform less efficiently. If this value is decreased, fewer events are processed per polling period and the adapter's performance might improve slightly.
Globalized	No
Bidi supported	No

Passphrase property (passPhrase)

This property is used for enhanced security by encrypting the private key.

Table 149. Passphrase property property characteristics

Required	No
Default	None
Property type	String
Usage	Used for enhanced security. It protects the private key by encrypting it in a SFTP configuration.
Globalized	Yes
Bidi supported	No

Port number property (PortNumber)

Port number of the FTP server through which the connection is established during inbound processing.

Table 150. Port number property characteristics

Required	Yes
Default	`21` for FTP and FTPS in Explicit mode, `990` for FTPS in Implicit mode, and `22` for SFTP.
Property type	Integer
Globalized	No

Private key file property (PrivateKeyFilePath)

This property enables you to browse and select the private key, which is used to authenticate to the Secure shell server.

Table 151. Private key property characteristics

Required	No
Default	None
Property type	String
Usage	Absolute path of the file which contains the private key. Used to authenticate the user to the Secure shell server.
Example	`c:\temp\key.ppk`
Globalized	Yes
Bidi supported	No

Protocol property (Protocol)

Protocol that determines whether the connection to be established is a normal FTP connection or a secure FTP connection.

For example:

Normal connection: `FTP`

FTP over SSL connection: `FTPS_SSL`

FTP over TLS connection: `FTPS_TLS`

FTP over SSH connection: `SFTP`

Table 152. Protocol property characteristics

Required	Yes
Default	`FTP`
Property type	String
Globalized	No

Remote archive directory property (ftpArchiveDirectory)

Relative path of the archive directory on the FTP server. The directory must exist. There are several options for using this property to specify archiving:

- Specifying a value for this property, but no value for the FTPRenameExt property causes the adapter to append a timestamp to the event file name and move it to the FTP server archive directory specified in this property.
- Specifying a value for this property and the FTPRenameExt property causes the adapter to rename the processed event file name with a timestamp and the value specified in FTPRenameExt and moves it to the FTP server archive directory specified in this property.
- Specifying no value either for this property or the FTPRenameExt property causes the adapter to delete the processed event file without archiving it.
- Specifying no value for this property but specifying a value for the FTPRenameExt property causes the adapter to rename the processed event file, adding a timestamp and the value specified in FTPRenameExt.

Table 153. Remote archive directory property characteristics

Required	No
Default	None
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the remote archive directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${REMOTEARCHIVE_DIRECTORY}`. See the topic on "Defining WebSphere Application Server environment variables" on page 71 in this documentation. The archive directory located on the FTP server and used in inbound configuration represents the absolute path of the archive directory. It does not contain any host name or URL information. This directory is located on the same FTP server where the Event Directory is located, for example: /home/archive. **Note:** The **FTPArchiveDirectory** has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Remote directory property (EventDirectory)

Remote directory of the FTP server from where the event files are retrieved for inbound processing.

Note: The value of event directory property should represent the absolute path of the directory.

Table 154. Remote directory property characteristics

Required	Yes
Default	None
Property type	String

Table 154. Remote directory property characteristics (continued)

Usage	You can use a WebSphere Application Server environment variable to represent the remote directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${REMOTE_DIRECTORY}`. See the topic on "Defining WebSphere Application Server environment variables" on page 71 in this documentation. **Note:** The **EventDirectory** has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Retrieve files with this pattern property (EventFileMask)

Filter for the event files. The file filter is a well-qualified expression consisting of alphanumeric characters and the * and ? wild cards.

Table 155. Retrieve files with this pattern property characteristics

Required	Yes
Default	`*.*`
Property type	String
Globalized	Yes

Enable remote verification property (enableRemoteVerification)

When a client connects to the FTP server, two kinds of connections or channels are established; a command connection (also known as control connection), and a data connection. The command connection is the one through which the FTP commands are sent (and replies to these commands received) to the server and the data connection is the channel through which the data transfer takes place between the client and the server.

This property is used to verify if the host system requesting the data transfer to or from the FTP server is the same host system on which the adapter is running.

The verification is done while establishing a data connection to perform data transfer.

Note: This property is applicable only to FTP and FTPS protocols.

Table 156. Enable Remote verification property characteristics

Required	No
Possible values	`True` `False`
Default	`True`
Property type	Boolean

Table 156. Enable Remote verification property characteristics (continued)

Usage	This property verifies if the data connection and the control connection are from the same host system. By default, the remote verification property is set to TRUE by the FTP server. When this property is set to: • True, during run time, the adapter checks if the data connection is established with the same host as the control connection. If the data connection is established from a different host than the control connection, then an exception is thrown and the connection fails. • False, remote verification is not performed. **Note:** Disabling the remote verification leads to low security. Precaution must be taken before disabling the remote verification.
Globalized	No
Bidi supported	No

Retry EIS connection on startup (RetryConnectionOnStartup)

This property controls whether the adapter attempts to connect again to the FTP server if it cannot connect at startup.

Table 157. Retry EIS connection on startup details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	This property indicates whether the adapter should retry the connection to the FTP server if the connection cannot be made when the adapter is started: • Set the property to `False` when you want immediate feedback about whether the adapter can establish a connection to the FTP server, for example, when you are building and testing the application that receives events from the adapter. If the adapter cannot connect, the adapter writes log and trace information and stops. The administrative console shows the application status as `Stopped`. After you resolve the connection problem, start the adapter manually. • Set the property to `True` if you do not need immediate feedback about the connection. If the adapter cannot connect during startup, it writes log and trace information, and then attempts to reconnect, using the RetryInterval property to determine how frequently to retry and the value of the RetryLimit property to retry multiple times until that value is reached. The administrative console shows the application status as `Started`.
Globalized	No
Bidi supported	No

Retry interval if connection fails (RetryInterval)

When the adapter encounters an error related to the inbound connection, this property specifies the length of time the adapter waits before trying to establish a new connection.

Table 158. Retry interval details

Required	Yes
Default	`2000`
Unit of measure	Milliseconds
Property type	Integer
Usage	Only positive values are valid. When the adapter encounters an error related to the inbound connection, this property specifies the length of time the adapter waits before trying to establish a new connection.
Globalized	No
Bidi supported	No

Number of times to retry the system connection (RetryLimit)

This property specifies the number of times the adapter tries to reestablish an inbound connection.

Table 159. Number of times to retry the system connection details

Required	No
Possible values	`0` and positive integers
Default	`0`
Property type	Integer
Usage	This property controls how many times the adapter retries the connection if the adapter cannot connect to the FTP server to perform inbound processing. A value of `0` indicates an infinite number of retries. To control whether the adapter retries if it cannot connect to the FTP server when it is first started, use the RetryConnectionOnStartup property.
Globalized	No
Bidi supported	No

Enable server verification property (EnableServerVerification)

This property is used to enable the remote server verification for SFTP protocol.

Table 160. Enable server verification property details

Required	No
Possible values	`True` `False`
Default	False
Property type	Boolean

Table 160. Enable server verification property details (continued)

Usage	When this property is set to: • True, server authentication is enabled • False, server authentication is disabled The adapter checks for the HostKeyFile property in the path of the file that contains the host keys of the trusted servers.
Globalized	Yes
Bidi supported	No

Host key file property (HostKeyFile)

This property provides the absolute path of the host key file that contains the host key of the trusted servers.

Table 161. Host key file property characteristics

Required	This property has to be specified if the EnableServerVerification property is enabled.
Default	None
Property type	String
Usage	This is used by the adapter to verify the host key of the remote server with the host keys of the trusted servers specified in this file.
Globalized	Yes
Bidi supported	No

Host name property (SocksProxyHost)

Host name of the machine used as a proxy server through which the adapter requests are routed to the FTP server.

Table 162. Host name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password property (SocksProxyPassword)

Password used to authenticate the proxy server.

Table 163. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Port number property (SocksProxyPort)

Port number of the proxy server through which the adapter requests are routed to the FTP server.

Table 164. Port number property characteristics

Required	No
Default	1080
Property type	Integer
Globalized	No

User name property (SocksProxyUserName)

User name used to authenticate the proxy server.

Table 165. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Sort event files property (SortEventFiles)

Determines the sorting order of event files being polled. Supported values are:

- by file name – sort ascending on file name
- by time stamp – sort ascending on last modified timestamp
- no sort – not sorted

Event file ordering from which events need to be delivered is valid only if the activation specification DeliveryType property is set to ORDERED. file name sorting is provided based on the locale of the FTP server. The ICU4J package is used to track the locales and their corresponding rules.

Table 166. Sort event files property characteristics

Required	No
Default	no sort (= not sorted)
Property type	String
Globalized	No

Specify criteria to split file content property (SplitCriteria)

This property takes different values based on the value of the SplittingFunctionClassName property. For example: To specify that a file is to be split every 5 KB, set the SplitCriteria property to 5000.

- If the SplittingFunctionClassName property specifies that files are split based on a delimiter, then SplitCriteria contains the delimiter that separates the business objects in the event file.
- If SplittingFunctionClassName is set to a value which does splitting based on size, then the SplitCriteria property contains a valid number that represents the size in bytes.

– If the event file size is greater than this value, the adapter splits the file into chunks of this size and the chunks are posted.
– If the event file size is less than this value, the entire event file is posted. When SplitCriteria=0, chunking is disabled.

When FilePassByReference is enabled during inbound PassThrough, the event file is not split.

Note: For input files that contain multiple COBOL copybook records, in order to enable file splitting by size you must provide the correct length of each record. To determine the size of each record, use one of these methods:

1. Open the Business Object in a text editor.
   a. For example:

```
<element name="CustomerNumber">
<annotation>
<appinfo source="http://www.ibm.com/cam/2005/typedescriptor">
<td:typeDescriptorElement>
<td:initialValue kind="SPACE"/>
<td:simpleInstanceTD accessor="readWrite" attributeInBit="false"
contentSize="5" offset="0" size="5">
<td:sharedType>
<td:stringTD addrUnit="byte" alignment="byte" characterSize="1"
lengthEncoding="fixedLength" paddingCharacter=" "
prefixLength="0" width="5"/>
</td:sharedType>
</td:simpleInstanceTD>
</td:typeDescriptorElement>
</appinfo>
</annotation>
<simpleType>
   <restriction base="string">
    <maxLength value="5"/>
  </restriction>
 </simpleType>
</element>
```

   Each element in the business object has a corresponding `<element>` entry.
   b. Look for a restriction tag for each element tag (the COBOL data binding requires a fixed-width data handler).
   c. Add up the `maxLength` attribute values for each of the elements. In this example, the value is 5. The sum of the `maxLength` values is the size of each record of type DFHCOMMAREA.
2. Open the Business Object in a text editor.
   a. Look for the complex type tag with the business object name value in the name attribute. In the example that follows, the business object name is DFHCOMMAREA.
   b. Locate a namespace-appended tag called `aggregateInstanceTD` and use the value for the attribute `contentSize`. In this example, the value is 117. This is the size of each record of type DFHCOMMAREA.

```
<complexType name="DFHCOMMAREA">
<annotation>
<appinfo source="http://www.ibm.com/cam/2005/typedescriptor">
<td:typeDescriptorCT>
<td:aggregateInstanceTD accessor="readWrite" attributeInBit="false"
contentSize="117" offset="0" size="117">
```

Table 167. Specify criteria to split file content property characteristics

Required	No

Table 167. Specify criteria to split file content property characteristics (continued)

Default	`0`
Property type	String
Globalized	Yes

Splitting function class name property

This value takes the fully qualified class name of the class file to be used to enable file splitting. Requires two values:

- The `com.ibm.j2ca.utils.filesplit.SplitByDelimiter` class that splits the event file based on delimiter.
- The `com.ibm.j2ca.utils.filesplit.SplitBySize` class that splits the event file based on the event file size.

Optionally, you can provide a custom file splitter class and use it by inputting the class name into the SplittingFunctionClassName property.

The delimiter or file size is provided in the SplitCriteria property. If the EventContentType property is set to null, it is automatically set to a class name that performs splitting based on file size.

Table 168. Splitting function class name property characteristics

Required	No
Default	`com.ibm.j2ca.utils.filesplit.SplitBySize`
Property type	String
Globalized	No

Stop the adapter when an error is encountered while polling (StopPollingOnError)

This property specifies whether the adapter will stop polling for events when it encounters an error during polling.

Table 169. Stop the adapter when an error is encountered while polling details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	If this property is set to `True`, the adapter stops polling when it encounters an error. If this property is set to `False`, the adapter logs an exception when it encounters an error during polling and continues polling.
Globalized	No
Bidi supported	No

Success file extension for local archive property (SuccessArchiveExt)

File extension used to archive all the successfully processed business objects. This property is used only when LocalArchiveDirectory is valid and exists. For example, `12345.order > 12345.order.success`

Table 170. Success file extension for local archive property characteristics

Required	No
Default	`success`
Property type	String
Globalized	Yes

User name property (UserName)

Name of the user who has privileges to connect to the FTP server and perform FTP operations. You do not need to specify a value for this property if the user name is included in the URL specified in the EventDirectory property.

Table 171. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

User name used to connect to event data source property (EP_UserName)

User name used by event persistence for getting the database connection from the data source.

Table 172. User name used to connect to event data source property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Rule editor to filter files

This property is used to filter event files based on a set of rules

Table 173. Rule editor to filter files

Required	Optional
Default	None
Property type	String
Usage	During an inbound processing, if the value in the rule table is specified, then the event files are fetched after filtering, based on the specified rules before polling those event files.
Globalized	Yes
Bidi supported	No

Globalization

WebSphere Adapter for FTP is a globalized application that can be used in multiple linguistic and cultural environments. Based on character set support and the locale of the host server, the adapter delivers message text in the appropriate language. The adapter supports bidirectional script data transformation between integration components.

Globalization and bidirectional transformation

The adapter is globalized to support single- and multi-byte character sets and deliver message text in the specified language. The adapter also performs bidirectional transformation, which refers to the task of processing data that contains both left-to-right (Hebrew or Arabic, for example), and right-to-left (a URL or file path, for example) semantic content within the same file.

Globalization

The Java runtime environment within the Java virtual machine (JVM) represents data in the Unicode character code set. Unicode contains encodings for characters in most known character code sets (both single- and multi-byte). Components in the WebSphere Business Integration system are written in Java. Therefore, when data is transferred between WebSphere Business Integration system components, there is no need for character conversion.

To log error and informational messages in the appropriate language and for the appropriate country or region, the adapter uses the locale of the system on which it is running.

Bidirectional transformation

Languages such as Arabic and Hebrew are written from right to left, yet they contain embedded segments of text that are written left to right, resulting in bidirectional script. When software applications handle bidirectional script, standards are used to display and process it. WebSphere Process Server and WebSphere Enterprise Service Bus use the Windows standard format, but an enterprise information system exchanging data with WebSphere Process Server or WebSphere Enterprise Service Bus can use a different format. WebSphere Adapters transform bidirectional script data passed between the two systems so that it is accurately processed and displayed on both sides of a transaction.

Bidirectional format

WebSphere Process Server and WebSphere Enterprise Service Bus use the bidirectional format of ILYNN (implicit, left-to-right, on, off, nominal). This is the format used by Windows. If an enterprise information system uses a different format, the adapter converts the format prior to introducing the data to WebSphere Process Server or WebSphere Enterprise Service Bus.

The bidirectional format consists of five attributes. When you set bidirectional properties, you assign values for each of these attributes. The attributes and settings are listed in the following table.

Table 174. Bidirectional format attributes

Letter position	Purpose	Values	Description	Default setting
1	Order schema	I	Implicit (Logical)	I
		V	Visual	
2	Direction	L	Left-to-Right	L
		R	Right-to-Left	
		C	Contextual Left-to-Right	
		D	Contextual Right-to-Left	
3	Symmetric Swapping	Y	Symmetric swapping is on	Y
		N	Symmetric swapping is off	
4	Text Shaping	S	Text is shaped	N
		N	Text is not shaped (Nominal)	
		I	Initial shaping	
		M	Middle shaping	
		F	Final shaping	
		B	Isolated shaping	
5	Numeric Shaping	H	National (Hindi)	N
		C	Contextual shaping	
		N	Numbers are not shaped (Nominal)	

The adapter transforms data into a logical, left-to-right format before sending the data to WebSphere Process Server or WebSphere Enterprise Service Bus.

Using bidirectional properties

You can use multiple bidirectional properties to control the transformation of both content data and metadata. You can set special bidirectional properties to exclude either content data or metadata from bidirectional transformation, or to identify data that requires special treatment during a transformation.

The following table describes four types of bidirectional properties.

Table 175. Bidirectional property types

Property type	Data transformations
EIS	Controls the format for content data, or data that is sent by the enterprise information system.
Metadata	Controls the format for metadata, or data that provides information about the content data.
Skip	Identifies content or metadata to exclude from transformation.

Table 175. Bidirectional property types (continued)

Property type	Data transformations
Special Format	Identifies certain text, such as file paths or URLs, that require different treatment during the transformation process. Can be set for either content data or metadata.

You can set properties that control bidirectional transformation in three areas.

- **Resource adapter properties:** These properties store default configuration settings, including the TurnBiDiOff property, which controls whether the adapter instance performs bidirectional transformation or not. Use the administrative console of the server to configure these properties.
- **Managed (J2C) connection factory properties:** These properties are used at run time to create an outbound connection instance with an enterprise information system. Once the managed connection factory properties are created, they are stored in the deployment descriptor.
- **Activation specification properties:** These properties hold the inbound event processing configuration information for a message endpoint. Set them as you perform external service, or use the administrative console of the server.

Business object annotations

Some adapters allow you to annotate bidirectional properties within a business object. Do this to add information that specifically controls the transformation of a business object or part of a business object. Use business object editor, a tool within WebSphere Integration Developer, to add annotations at these levels:

- Business object
- Business object application-specific attribute
- Business object attribute
- Business object attribute application-specific attribute

Property scope and lookup mechanism

After you set values for bidirectional properties for an adapter and annotate business objects where appropriate, the adapter performs bidirectional transformations. It does so by using logic that relies on a hierarchical inheritance of property settings and a lookup mechanism.

Properties defined within the resource adapter are at the top of the hierarchy, while those defined within other areas or annotated within a business object are at lower levels of the hierarchy. So for example, if you only set values for EIS-type bidirectional properties for the resource adapter, those values are inherited and used by transformations that require a defined EIS-type bidirectional property whether they arise from an inbound (activation specification) transaction or an outbound (managed connection factory) transaction.

However, if you set values for EIS-type bidirectional properties for both the resource adapter and the activation specification, a transformation arising from an inbound transaction uses the values set for the activation specification.

The processing logic uses a lookup mechanism to search for bidirectional property values to use during a transformation. The lookup mechanism begins its search at the level where the transformation arises and searches upward through the

hierarchy for defined values of the appropriate property type. It uses the first valid value it finds. It searches the hierarchy from child to parent only; siblings are not considered in the search.

Bidirectional transformation in business objects

For outbound processing, you can modify the business objects to enable the bidirectional transformation of the wrapper properties in the WebSphere Adapter for FTP business object and the data in content-specific or generic business objects.

You have to add an annotation to the complex type of the business object to specify the bidirectional formatting attributes in the files for the following business objects:

- For the generic business object, change the FTPFile.xsd file.
- For the user-defined business object, change the customer wrapper (for example, the CustomWrapper.xsd file and Customer.xsd).
- For the UnstructuredContent business object, change the UnstructuredContent.xsd.

The following sections include annotations that can serve as examples.

Bidirectional formatting attributes of the business object

The following annotation, which contains the bidirectional context information, applies to all the attributes in the FTP business objects. The FTPFileBaseDataBinding uses the bidirectional information in the element BiDiContext to transform all the attributes.

```
<xsd:complexType name="Customer">
<xsd:annotation>
            <xsd:appinf
                source="http://www.ibm.com/xmlns/prod/websphere/j2ca/datatrans
formation/databindingm
apping">
                <dtm:DataBindingMapping
                    xsi:type="dtm:DataBindingMapping"
                    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                    xmlns:dtm="http://www.ibm.com/xmlns/prod/websphere/j2ca/da
tatransformation/databindingmapping">
                    <BiDiContext>
                            <orientation>rtl</orientation>
                            <textShape>nominal</textShape>
                            <orderingScheme>visual</orderingScheme>
                            <symmetricSwapping>true</symmetricSwapping>
                            <numeralShapes>nominal</numeralShapes>
                    </BiDiContext>
                </dtm:DataBindingMapping>
            </xsd:appinfo>
        </xsd:annotation>
```

Bidirectional formatting attributes of the wrapper

You can add an annotation to the wrapper of a user-defined type business object. The annotation in the wrapper business objects such as the generic (FTPFile) and the user-defined type (CustomerWrapper) is used to do bidirectional transformation of wrapper attributes. The content-specific business objects that are used inside the wrapper business objects are not transformed using annotation in the wrapper business objects. To transform content-specific business objects, you

must edit the respective business object definition to add the annotation shown in the previous example for bidirectional formatting of attributes of the business object.

The following annotation is an example for the wrapper:

```
<complexType name="CustomerWrapper">
<annotation>
            <appinfo
                source="http://www.ibm.com/xmlns/prod/websphere/j2ca/
datatransformation/databindingmapping">
                <dtm:DataBindingMapping
                    xsi:type="dtm:DataBindingMapping"
                    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                    xmlns:dtm="http://www.ibm.com/xmlns/prod/websphere/j2ca/
datatransformation/databindingmapping">
                    <BiDiContext>
                            <orientation>rtl</orientation>
                            <textShape>nominal</textShape>
                            <orderingScheme>visual</orderingScheme>
                            <symmetricSwapping>true</symmetricSwapping>
                            <numeralShapes>nominal</numeralShapes>
                    </BiDiContext>
                </dtm:DataBindingMapping>
            </appinfo>
        </annotation>
```

Properties enabled for bidirectional data transformation

Bidirectional data transformation properties enforce the correct format of bidirectional script data exchanged between an application or file system and integration tools and runtime environments. After these properties are set, bidirectional script data is correctly processed and displayed in WebSphere Integration Developer and WebSphere Process Server or WebSphere Enterprise Service Bus.

Managed (J2C) connection factory properties

The following managed (J2C) connection properties control bidirectional transformation.

- Username
- Password
- Directory
- FileName
- StagingDirectory
- SecondServerUsername
- SecondServerPassword
- SecondServerDirectory
- SocksProxyUsername
- SocksProxyPassword
- FileSequenceLog

Activation specification properties

The following activation specification properties control bidirectional transformation.

- Username

- Password
- EventDirectory
- EventFileMask
- FTPArchiveDirectory
- LocalEventDirectory
- LocalArchiveDirectory
- FTPScriptFileExecutedBeforeInbound
- FTPScriptFileExecutedAfterInbound
- FTPRenameExt
- FailedArchiveExt
- OriginalArchiveExt
- SuccessArchiveExt
- SocksProxyUsername
- SocksProxyPassword

Deployment Descriptor configuration properties

The following Deployment Descriptor configuration properties control bidirectional transformation.

- EPDataSourceJNDIName
- EPEventTableName
- EPDatabaseUsername
- EPDatabasePassword
- EPDatabaseSchemaName

Wrapper business object properties

The following wrapper business object properties control bidirectional transformation.

- DirectoryPath
- Filename
- FtpServerEventDirectory
- SecondServerDirectory
- SecondServerUsername
- SecondServerPassword
- LocalDirectoryPath
- LocalArchiveDirForCreate
- StagingDirectory
- ArchiveDirectoryForRetrieve

Adapter messages

View the messages issued by WebSphere Adapter for FTP at the following location.

Link to messages: http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/topic/com.ibm.wbit.help.messages.doc/messages.html

The displayed Web page shows a list of message prefixes. Click a message prefix to see all the messages with that prefix:

- Messages with the prefix CWYFT are issued by WebSphere Adapter for FTP
- Messages with the prefix CWYBS are issued by the adapter foundation classes, which are used by all the adapters

Related information

The following information centers, IBM Redbooks, and Web pages contain related information for the WebSphere Adapter for FTP.

Samples and tutorials

To help you use WebSphere Adapters, samples and tutorials are available from the Business Process Management Samples and Tutorials Web site. You can access the samples and tutorials in either of the following ways:

- From the welcome page that opens when you start WebSphere Integration Developer. To see samples and tutorials for WebSphere Adapter for FTP, click **Retrieve**. Then browse the displayed categories to make your selections.
- At this location on the Web: http://publib.boulder.ibm.com/bpcsamp/index.html.

Information resources

- The WebSphere Business Process Management information resources Web page includes links to articles, Redbooks, documentation, and educational offerings to help you learn about WebSphere Adapters: http://www14.software.ibm.com/webapp/wsbroker/redirect?version=pix&product=wps-dist&topic=bpmroadmaps
- The WebSphere Adapters library page includes links to all versions of the documentation: http://www.ibm.com/software/integration/wbiadapters/library/infocenter/

Information about related products

- WebSphere Business Process Management, version 6.2.x, information center, which includes WebSphere Process Server, WebSphere Enterprise Service Bus, and WebSphere Integration Developer information: http://publib.boulder.ibm.com/infocenter/dmndhelp/v6r2mx/index.jsp
- WebSphere Adapters, version 6.1.x, information center: http://publib.boulder.ibm.com/infocenter/dmndhelp/v6r1mx/index.jsp
- WebSphere Adapters, version 6.0, information center: http://publib.boulder.ibm.com/infocenter/wbihelp/v6rxmx/topic/com.ibm.wsadapters.doc/welcome_wsa.html
- WebSphere Business Integration Adapters information center: http://publib.boulder.ibm.com/infocenter/wbihelp/v6rxmx/index.jsp?topic=/com.ibm.wbi_adapters.doc/welcome_adapters.htm

developerWorks® resources

- WebSphere Adapter Toolkit
- WebSphere business integration zone

Support and assistance

- WebSphere Adapters technical support: http://www.ibm.com/software/integration/wbiadapters/support/
- WebSphere Adapters technotes: http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8

&dc=DB520+D800+D900+DA900+DA800+DB560&dtm. In the **Product category** list, select the name of the adapter and click **Go**.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Department 2Z4A/SOM1
294 Route 100
Somers, NY 10589-0100
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows: (c) (your company name) (year). Portions of

this code are derived from IBM Corp. Sample Programs. (c) Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming interface information

Programming interface information, if provided, is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Warning:

Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks and service marks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A complete and current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

This product includes software developed by the Eclipse Project (http://www.eclipse.org).

Index

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W

X

Printed in USA