webMethods Tuxedo Adapter Installation and User's Guide

Version 6.0

July 2012

WEBMETHODS

This document applies to webMethods Tuxedo Adapter 6.0 and to all subsequent releases.

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

Document ID: ADAPTER-TUX-IUG-60-20210331

Table of Contents

About this Guide

This guide describes how to configure and use the webMethods Tuxedo Adapter. It contains information for administrators and application developers who want to exchange data with Tuxedo systems.

To use this guide effectively, you should:

- Be familiar with BEA Jolt and BEA Tuxedo servers.
- Be familiar with the terminology and basic operations of your operating system.
- Be familiar with the setup and operation of the webMethods Integration Server.
- Have a general idea about how to perform basic tasks with Digital Event Services or Software AG Designer.

> **Note:**
> Procedures for creating flow services, adapter notifications, and adapter services are similar in Digital Event Services and Designer.

Document Conventions

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

Online Information and Support

Software AG Documentation Website

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

Software AG Empower Product Support Website

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

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

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

To submit feature/enhancement requests, get information about product availability, and download products, go to Products.

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

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

Software AG TECHcommunity

You can find documentation and other technical information on the Software AG TECHcommunity website at http://techcommunity.softwareag.com. You can:

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

Data Protection

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

1 Overview of the webMethods Tuxedo Adapter

About the Adapter

The webMethods Tuxedo Adapter is an add-on to the webMethods Integration Server that enables you to exchange data between an Integration Server and a BEA Jolt server, an interface to the BEA Tuxedo application server. The Tuxedo Adapter provides seamless and real-time communication with the application server.

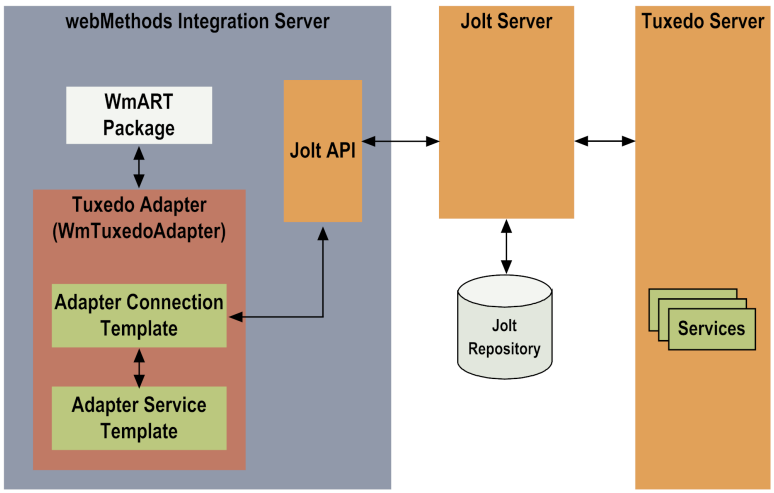
The Tuxedo Adapter provides a service template that enables you to configure and manage adapter connections and adapter services. The adapter services are linked to specific Tuxedo services and can be used on their own or included in flows or Java services built to implement business processes.

Using functionality provided by the adapter, a client can connect to the Jolt server and invoke Tuxedo services. For example, you can use the Tuxedo Adapter to create an adapter service that updates account information by calling a service in a Tuxedo application.

Architectural Overview

The Tuxedo Adapter provides a set of user interfaces, services, and a template that enable you to create integrations with Tuxedo applications. The adapter is provided as a single package that must be installed on the Integration Server. For detailed installation instructions, see "Installing and Uninstalling the Tuxedo Adapter " on page 21. For software requirements, see the *webMethods Adapters System Requirements* .

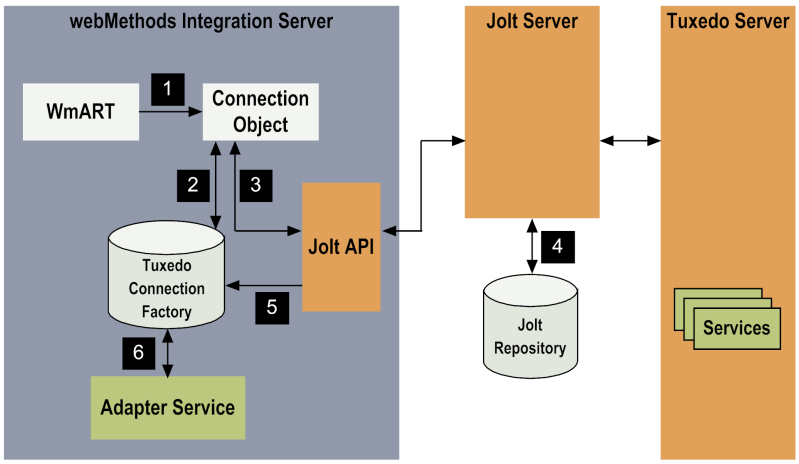
The following diagram illustrates how the Tuxedo Adapter interfaces with a Tuxedo system using the Jolt APIs. Following the diagram are descriptions of the architectural components involved in the integration process.



- **Integration Server** . The Tuxedo Adapter is installed and runs on the Integration Server.
- **WmART Package**. The WmART package provides a common framework for Integration Server adapters to use the Integration Server functionality, making the Integration Server the run-time environment for the Tuxedo Adapter. The WmART package is installed with the Integration Server and provides logging, transaction management, and error handling for the adapter and its connections and services.

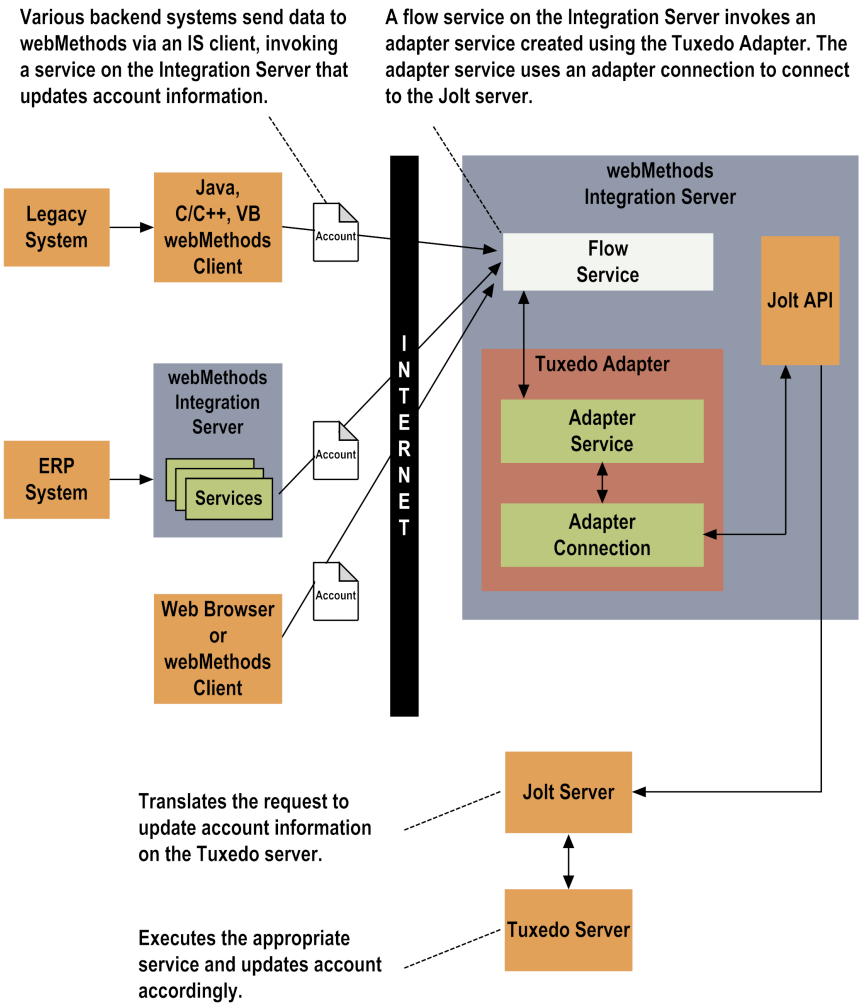
- **Tuxedo Adapter** . The Tuxedo Adapter is delivered as a single package called WmTuxedoAdapter. The adapter provides Integration Server Administrator user interfaces that enable you to configure and manage adapter connections, and Digital Event Services and Software AG Designer user interfaces that enable you to configure and manage adapter services. The Tuxedo Adapter installation includes templates from which you can create Tuxedo Adapter connections and services.
- **Adapter Connection Template**. An adapter connection enables the Integration Server to connect to a Jolt server at run time. You must configure an adapter connection before you can create adapter services. The Tuxedo Adapter provides a template for adapter connections in the Integration Server Administrator. For a detailed description of adapter connections and usage information, see "Adapter Connections" on page 14.
- **Adapter Service Template**. An adapter service enables the Integration Server to execute Tuxedo services. For example, an adapter service can enable Integration Server clients to write account information to an employee database connected to the Tuxedo server. The Tuxedo Adapter provides an adapter service template in the Digital Event Services and Designer. For more information about the adapter service template and how the services interact, see "Adapter Services" on page 16.
- **Jolt API**. The Tuxedo Adapter implements the Java-based Jolt API to connect to the Jolt server and initiate a service on the Tuxedo server.
- **Jolt Server.** The Jolt server acts as a proxy by passing service requests to and from the Tuxedo server on behalf of the Tuxedo Adapter.
- **Jolt Repository**. The Jolt repository contains all Tuxedo service names and input/output signatures as well as the associated data type information. After the Integration Server makes the initial connection, the Tuxedo Adapter retrieves applicable service definitions from the Jolt repository. The Integration Server stores these definitions in a cache known as the TuxedoConnectionFactory. For more information, see the diagram that follows.
- **Tuxedo Server**. The Tuxedo server receives requests to execute services from Tuxedo applications.

The basic data flow from Integration Server to the Jolt server is as follows:



Step	Action
1	Using the Integration Server Administrator, you configure a connection in which the WmART package creates a connection object.
2	When the connection is instantiated, the connection object first checks the TuxedoConnectionFactory for the Tuxedo service definitions. If the definitions are not present, the Integration Server proceeds to step 3. If the definitions are present, the Integration Server skips to step 6.
3	The Integration Server implements the Jolt API to connect to the Jolt server.
4	The Jolt server returns the following information from its repository: ■ Names of all Tuxedo/Jolt services that are associated with this connection, based on the user login, role, and application access. ■ Input and output parameters for each service. ■ Maximum number of occurrences of each input and output parameter.
5	The Integration Server stores the information in the tables (cache) associated with the connection. **Note:** If you need to update this information, disable and re-enable the connection. For instructions, see "Adapter Connections" on page 33.
6	The Tuxedo Adapter accesses the information in the TuxedoConnectionFactory when you create an adapter service. You configure adapter services using a template provided with the Tuxedo Adapter. The template provides all of the code necessary for interacting with the Jolt server, without the data specifications.

The following diagram illustrates the Tuxedo Adapter and the webMethods Integration Server in a business-process integration.



Adapter Package Management

The Tuxedo Adapter is provided as a package called WmTuxedoAdapter that you manage like any package on the Integration Server.

There are several considerations regarding how you set up and effectively manage your packages on the Integration Server, such as those described in the following list.

- Create user-defined packages for your connections and adapter services. For details, see "Managing the Adapter Package" on page 26.
- Understand how package dependencies work so that you make the best decisions about how to manage your adapter services. For details, see"Package Dependency Requirements and Guidelines" on page 26.
- Control which development groups have access to which adapter services. For details, see "Controlling Group Access" on page 28.

- Understand how clustering, an advanced feature of the webMethods Integration Server, works to effectively manage your adapter services. For details, see "Configuring the Adapter in a Clustered Environment" on page 29.

Adapter Connections

Tuxedo Adapter connections contain parameters that the Integration Server uses to manage connections to the Jolt server, using the Jolt API. The Tuxedo Adapter then uses these connections to provide adapter services. You can configure connections using the Integration Server Administrator (you must have Integration Server administrator privileges to access the Tuxedo Adapter administrative screens).

Tuxedo Adapter connections are based on one connection template. This template creates connections that can be used in non-distributed, local transactions.

You create one or more connections at design time to use in integrations. The number of connections you create depend on your integration needs. For example, if you are invoking services from different Tuxedo applications, you might require different user names and passwords for each application, based on your application access and role.

For instructions to configure, view, edit, enable, and disable Tuxedo Adapter connections, see "Adapter Connections" on page 33. For information about setting user privileges, see the *webMethods Integration Server Administrator's Guide* for your release.

For a list of tasks that you must complete before you can create your connections, see "Before Creating Adapter Connections" on page 34.

Connection Pools

The Integration Server includes a connection management service that dynamically manages connections and connection pools based on configuration settings that you specify for the connection. All adapter services use connection pooling.

A connection pool is a collection of connections with the same set of attributes. The Integration Server maintains connection pools in memory. Connection pools improve performance by enabling adapter services to reuse open connections instead of opening new connections.

Run-Time Behavior of Connection Pools

When you enable a connection, the Integration Server initializes the connection pool, creating the number of connection instances you specified in the connection's **Minimum Pool Size** field. Whenever an adapter service needs a connection, the Integration Server provides a connection from the pool. If no connections are available in the pool, and the maximum pool size has not been reached, the server creates one or more new connections (according to the number specified in **Pool Increment Size**) and adds them to the connection pool. If the pool is full (as specified in **Maximum Pool Size**), the requesting service will wait for the Integration Server to obtain a connection, up to the length of time specified in the **Block Timeout** field, until a connection becomes available. Periodically, the Integration Server inspects the pool and removes inactive

connections that have exceeded the expiration period that you specified in **Expire Timeout**. For information about configuring connections, see "Adapter Connections" on page 33.

Built-In Services For Connections

The Integration Server provides built-in services that enable you to programmatically control connections. You can use them to enable and disable a connection, and to return usage statistics, the current state (Enabled or Disabled), and error status for a connection. These services are located in the WmART package, in the pub.art.connection folder.

For details about built-in services, see the *webMethods Integration Server Built-In Services Reference* for your release.

Transaction Management of Adapter Connections

When you define a connection, the transaction type you choose determines the type of transaction management that the connection's operations will use. The Tuxedo Adapter supports transactional connection types. For a detailed discussion of transaction management and the Tuxedo Adapter, see "Transaction Management Overview" on page 68.

A transactional connection uses syncpoint processing, which means that you can group one or more requests into a single Logical Unit of Work (LUW). You can control these requests manually (explicit transactions), or you can allow the Integration Server's transaction manager control them for you (implicit transactions).

To control these requests explicitly within a given LUW, you use the built-in services described in "Built-In Transaction Management Services" on page 67. If you do not use these built-in services, the Integration Server will manage the requests automatically (implicitly).

If a transaction uses only a single connection, or multiple connections that are all from the same connection pool, you do not need to manage the transaction explicitly. However, if a transaction uses multiple LOCAL_TRANSACTION transaction connections from different connection pools, you must manage the transaction explicitly.

Note:
Implicit transactions complete when the flow service that contains the LUW finishes execution. If you create a looping operation within your LUW that could potentially involve a large number of requests, consider managing the transactions explicitly to reduce the possibility that you will need to roll back a large number of requests because a single request fails.

Changing the Connection Associated with an Adapter Service at Design Time

The Integration Server provides a built-in service that you can use at design time to change the connection associated with an adapter service. The pub.art.service:setAdapterServiceNodeConnection service in the WmART package enables you to change the connection. Using this service, you can change the specific connection associated with an adapter service at design time so that you do not need to create and maintain multiple adapter services.

Note:
You should run the pub.art.service:setAdapterServiceNodeConnection service at design time only. Do not use it within an Integration Server flow or Java service that also calls the adapter service whose associated connection you intend to change. You must run this service directly from the Digital Event Services or Designer by invoking it from a flow service and running the flow service. For more information, see the *webMethods Integration Server Built-In Services Reference* for your release.

Changing the Connection Associated with an Adapter Service at Run Time

The Integration Server enables you to dynamically select the connection a service uses to interact with the adapter's resources such as the Tuxedo server. This feature enables one service to interact with multiple resources.

For example, you can configure an adapter service to use a default connection that interacts with a production application server. However, at run time you can override the default connection and use another connection to interact with a test application server.

For more information about overriding a service's default connection at run time, see "Dynamically Changing Adapter Connections at Run Time" on page 37.

Adapter Services

Adapter services allow you to connect to the Jolt server and initiate a service on the Tuxedo server from the Integration Server.

You call adapter services from a flow or Java service to interact with the Jolt server. At design time, the adapter obtains service information about each application deployed on the Tuxedo server from the Jolt server. The adapter gets this information directly from the Jolt repository and stores it in the TuxedoConnectionFactory. From this information, you configure adapter services. The Integration Server then uses adapter connections that you defined earlier to execute the adapter services.

You configure adapter services using the template provided with the Tuxedo Adapter. The template represents a specific technique for doing work on the Jolt server, such as creating an account in an Tuxedo application. An adapter service template contains all the code necessary for interacting with the Jolt server but without the data specifications. You provide these specifications when you configure a new adapter service.

You use Digital Event Services or Designer to configure an adapter service. For more information, see the *webMethods Service Development Help* for your release.

Configuring a new adapter service from an adapter service template is straightforward. Using Digital Event Services or Designer, you assign the service an adapter connection. After you select the connection for the adapter service, you select the adapter service template. At this point, the adapter automatically populates the Tuxedo-specific fields in the adapter service editor tabs. You then use the editor to manipulate these fields to configure the adapter service.

The fields in the input and output signatures for an adapter service template are predefined and depend on the specific signatures of the Tuxedo service selected. The Integration Server parses Tuxedo fields that have more than one occurrence in the same Tuxedo application as an array of elements.

Important: All variables in the input signature will be rendered as java.lang.Strings and will appear as strings on the input pipeline regardless of their data type in Tuxedo. The execute() function of the Tuxedo Adapter will translate these strings to the proper input data types before passing them to Tuxedo.

Each adapter service you configure can be used as a standalone, executable entity. However, you can also incorporate your adapter services in flow services or Java services to implement business workflow. Using the Flow Service Editor in Digital Event Services or Designer, you can create intelligent business applications wrapped around the adapter services.

Adapter Service Template

The Tuxedo Adapter provides the synchronous adapter service template. The synchronous service adapter service type executes specified services on the Tuxedo server.

See "Configuring Synchronous Services" on page 44 for instructions.

Using Adapter Services

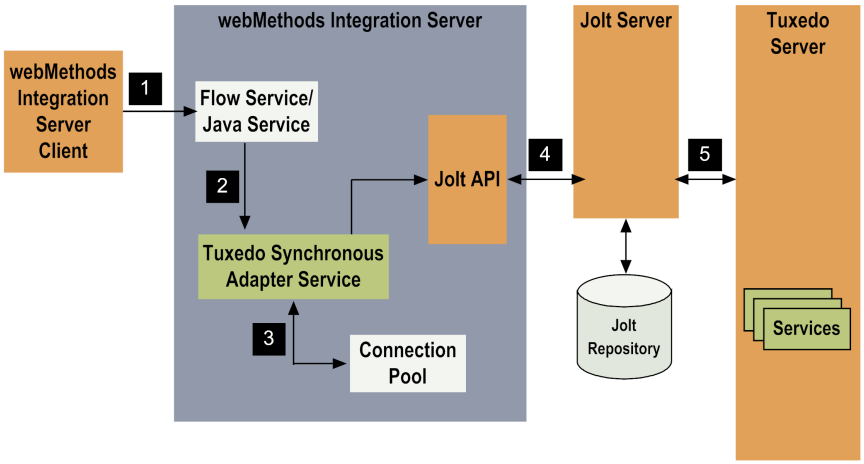
The following table lists the tasks required to use adapter services:

Step	Task	Use this tool...
1	Create an adapter connection. For details, see "Adapter Connections" on page 33.	Integration Server Administrator
2	Create a new adapter service, select the Tuxedo Adapter service template, and specify the Tuxedo Adapter connection. For more information about configuring adapter services, see "Configuring Synchronous Services" on page 44.	Digital Event Services or Designer
3	If you plan to use an Integration Server flow or Java service to invoke the adapter service, design the flow or Java service to use this adapter service. For details, see the *webMethods Service Development Help* for your release.	Digital Event Services or Designer

Step	Task	Use this tool...
4	Manage the adapter service. For details, see "Adapter Package Management" on page 25 and "Adapter Logging and Exception Handling" on page 51.	Digital Event Services or Designer and Integration Server Administrator

Adapter Service Transaction Processing

The following diagram illustrates how the Tuxedo Adapter processes adapter services at run time.



Step	Description
1	An Integration Server client runs a flow or Java service on the Integration Server.
2	The flow or Java service invokes a Tuxedo synchronous adapter service. You configured the adapter service earlier using the Digital Event Services or Designer.
3	The adapter service retrieves a connection from the connection pool. Adapter connections contain connection information for the Jolt server, including the Tuxedo application parameters.
4	The adapter service uses the Jolt API to connect to the Jolt server. You created and enabled the adapter connection earlier using the Integration Server Administrator.
5	The Jolt server translates the service request and submits the request to the Tuxedo server to execute the service.

Using Version Control Systems to Manage Adapter Elements

The adapter supports the Version Control System (VCS) Integration feature provided by Designer. When you enable the feature in Integration Server, you can check adapter packages or elements into and out of your version control system from Designer. For more information about the VCS Integration feature, see the *Configuring the VCS Integration Feature*.

Beginning with Integration Server 8.2 SP3, the adapter supports the local service development feature in Designer. This feature extends the functionality of the VCS Integration feature to check package elements and their supporting files into and out of a VCS directly from Designer. For more information about local service development and how it compares to the VCS Integration feature, see the *webMethods Service Development Help*.

Optimize Infrastructure Data Collector Support for the Adapter

Optimize Infrastructure Data Collector monitors the system and operational data associated with Integration Server run-time components such as Integration Servers, Broker Servers, Brokers, and adapters, and reports the status of these components on Optimize for Infrastructure or other external tools. When you start monitoring an Integration Server, Infrastructure Data Collector automatically starts monitoring all ART-based adapters that are installed on the Integration Server.

For information about monitored key performance indicators (KPIs) collected for the monitored adapter components, see the *Administering webMethods Optimize* and *webMethods Optimize User's Guide* for your release.

Viewing the Adapter's Update Level

Beginning with Integration Server 6.5, you can view the list of updates that have been applied to the adapter. The list of updates appears in the **Updates** field on the adapter's About page in the Integration Server Administrator.

Controlling Pagination

When using the adapter on Integration Server 8.0 and later, you can control the number of items that are displayed on the adapter Connections screen and Notifications screen. By default, 10 items are displayed per page. Click **Next** and **Previous** to move through the pages, or click a page number to go directly to a page.

To change the number of items displayed per page, set the watt.art.page.size property and specify a different number of items.

To set the number of items per page

1. From Integration Server Administrator, click **Settings > Extended**.

2. Click **Edit Extended Settings**. In the Extended Settings editor, add or update the watt.art.page.size property to specify the preferred number of items to display per page. For example, to display 50 items per page, specify:

```
watt.art.page.size=50
```

3. Click **Save Changes**. The property appears in the Extended Settings list.

4. For more information about working with extended configuration settings, see the *webMethods Integration Server Administrator's Guide* for your release.

2 Installing and Uninstalling the Tuxedo Adapter

Overview

This chapter explains how to install, upgrade, and uninstall webMethods Tuxedo Adapter 6.0. The instructions use the Software AG Installer and the Software AG Uninstaller wizards. For complete information about the wizards or other installation methods, or to install other Integration Server products, see the *Installing webMethods Products On Premises* for your release.

Requirements

For a list of the operating systems, Tuxedo products, and Integration Server products supported by the adapter, see the *webMethods Adapters System Requirements* .

Tuxedo Adapter 6.0 has no hardware requirements beyond those of its host Integration Server.

Installing Tuxedo Adapter 6.0

Note:
If you are installing the adapter in a clustered environment, you must install it on each Integration Server in the cluster, and each installation must be identical. For more information, see "Using Tuxedo Adapter in a Clustered Environment" on page 28.

To install Tuxedo Adapter 6.0

1. Download Software AG Installer from the Empower Product Support Web site.
2. If you are installing the adapter on an existing Integration Server, shut down Integration Server.
3. Start the Installer wizard.
4. Choose the Integration Server release that includes the Integration Server on which to install the adapter. For example, if you want to install the adapter on Integration Server 7.1, choose the 7.1 release.
5. Specify the installation directory as follows:
   - If you are installing on an existing Integration Server, specify the Integration Server installation directory that contains the host Integration Server.
   - If you are installing both the host Integration Server and the adapter, specify the installation directory to use.

     Installer will install the adapter in the *Integration Server_directory* \packages directory.
6. In the product selection list, select **Adapters > webMethods Tuxedo Adapter 6.0**. You can also choose to install documentation.

7. Go to the *Tuxedo_directory*\udataobj\jolt directory and copy the jolt.jar, joltadmin.jar, and joltjse.jar to the *Integration Server_directory* \packages\WmTuxedoAdapter\ code\jars directory. (If that directory does not exist in the Integration Server directory, create it.)

Uninstalling Tuxedo Adapter 6.0

To uninstall Tuxedo Adapter 6.0

1. Shut down the host Integration Server. You do not need to shut down any other Integration Server products or applications that are running on your machine.

2. Start Software AG Uninstaller, selecting the Integration Server installation directory that contains the host Integration Server. In the product selection list, select **Adapters > webMethods Tuxedo Adapter 6.0**. You can also choose to uninstall documentation.

3. Restart the host Integration Server.

4. Uninstaller removes all Tuxedo Adapter 6.0-related files that were installed. However, Uninstaller does not delete files created after you installed the adapter (for example, user-created or configuration files), nor does it delete the adapter directory structure. You can go to the *Integration Server_directory* \packages directory and delete the WmTuxedoAdapter directory.

3 Adapter Package Management

Overview

The following sections describe how to set up and manage your webMethods Tuxedo Adapter packages, set up Access Control Lists (ACL), and use the adapter in a clustered environment.

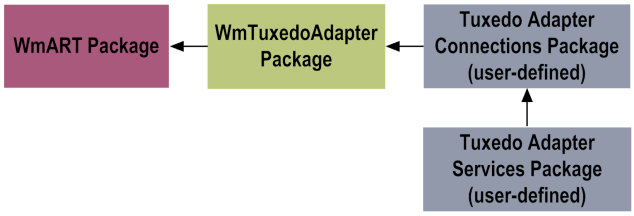
Managing the Adapter Package

The Tuxedo Adapter is provided as a package called WmTuxedoAdapter. You manage the WmTuxedoAdapter package as you would manage any package on the Integration Server.

When you create connections and adapter services, define them in user-defined packages rather than in the WmTuxedoAdapter package. Doing so allows you to manage the package more easily.

As you create packages in which to store connections and adapter services use the package management functionality provided in Digital Event Services or Software AG Designer and set the packages to have a dependency on the WmTuxedoAdapter package. That way, when the WmTuxedoAdapter package loads or reloads, the packages that you created load automatically.

The following diagram illustrates the dependencies for the Tuxedo Adapter.



Package management tasks include:

- Setting package dependencies (see "Package Dependency Requirements and Guidelines" on page 26).
- "Enabling Packages" on page 27.
- "Disabling Packages" on page 28.
- "Controlling Group Access" on page 28.

Package Dependency Requirements and Guidelines

This section contains a list of dependency requirements and guidelines for user-defined packages. For instructions about setting package dependencies, see the *webMethods Service Development Help* for your release.

- A user-defined package must have a dependency on its associated adapter package, WmTuxedoAdapter (the WmTuxedoAdapter package has a dependency on the WmART package).
- Package dependencies ensure that at startup the Integration Server automatically loads or reloads all packages in the proper order: the WmART package first, the adapter package next,

and the user-defined packages last. The WmART package is installed automatically when you install the Integration Server. You should not need to manually reload the WmART package.

- If the connections and adapter services of an adapter are defined in different packages, then:
  - A package that contains the connections must have a dependency on the adapter package.
  - Packages that contain adapter services must have a dependency on their associated connection package.
- Keep connections for different adapters in separate packages so that you do not create interdependencies between adapters. If a package contains connections for two different adapters, and you reload one of the adapter packages, the connections for both adapters will reload automatically.
- The Integration Server will not allow you to enable a package if it has a dependency on another package that is disabled. That is, before you can enable your package, you must enable all packages on which your package depends. For information about enabling packages, see"Enabling Packages" on page 27.
- The Integration Server *will* allow you to disable a package even if another package that is enabled has a dependency on it. Therefore, you must manually disable any user-defined packages that have a dependency on the adapter package before you disable the adapter package. For information about disabling packages, see "Disabling Packages" on page 28.
- You can name connections and adapter services the same name provided that they are in different folders and packages.

Enabling Packages

All packages are enabled automatically by default. Enabling an adapter package will not cause its associated user-defined package(s) to be reloaded. For information about reloading packages, see the *webMethods Service Development Help* for your release.

Note:
Before you manually enable a user-defined package, you must first enable its associated adapter package (WmTuxedoAdapter).

To enable a package

1. Open the Integration Server Administrator if it is not already open.
2. In the **Packages** menu of the navigation area, click **Management**.
3. Click **No** in the **Enabled** column. The server displays a ✔ and **Yes** in the **Enabled** column.

Disabling Packages

When you want to temporarily prohibit access to the elements in a package, disable the package. When you disable a package, the server unloads all of its elements from memory. Disabling a package prevents the Integration Server from loading that package at startup. A disabled package will remain disabled until you explicitly enable it using the Integration Server Administrator and will not be listed in Digital Event Services or Designer.

Note:
If your adapter has multiple user-defined packages, and you want to disable some of them, disable the adapter package first. Otherwise, errors will be issued when you try to access the remaining enabled user-defined packages.

To disable a package

1. Open the Integration Server Administrator if it is not already open.

2. In the **Packages** menu of the navigation area, click **Management**.

3. Click **Yes** in the **Enabled** column for the package that you want to disable. The server issues a prompt to verify that you want to disable the package. Click **OK** to disable the package. When the package is disabled, the server displays **No** in the **Enabled** column.

Controlling Group Access

To control which development group has access to which adapter services, use access control lists (ACLs). You can use ACLs to prevent one development group from inadvertently updating the work of another group, or to allow or deny access to services that are restricted to one group but not to others.

For general information about assigning and managing ACLs, see the *webMethods Service Development Help* for your release.

Using Tuxedo Adapter in a Clustered Environment

What is webMethods Integration Server Clustering?

Clustering is an advanced feature of the webMethods product suite that substantially extends the reliability, availability, and scalability of Integration Server. Clustering accomplishes this by providing the infrastructure and tools to deploy multiple Integration Servers as if they were a single virtual server and to deliver applications that leverage that architecture. Because this activity is transparent to the client, clustering makes multiple servers look and behave as one.

For details about Integration Server clustering, see the *webMethods Integration Server Clustering Guide* for your release.

Integration Server 8.2 SP2 and higher supports the caching and clustering functionality provided by Terracotta. Caching and clustering are configured at the Integration Server level and Tuxedo Adapter uses the caching mechanism that is enabled on Integration Server.

With clustering, you get the following benefits:

- **Load balancing.** This feature, provided automatically when you set up a clustered environment, allows you to spread the workload over several servers, thus improving performance and scalability.
- **Failover support.** Clustering enables you to avoid a single point of failure. If a server cannot handle a request, or becomes unavailable, the request is redirected automatically to another server in the cluster.

  Note:webMethods Integration Server clustering redirects HTTP and HTTPS requests, but does not redirect FTP or SMTP requests.
- **Scalability.** You can increase your capacity even further by adding new machines running Integration Server to the cluster.

Configuring the Adapter in a Clustered Environment

When you configure the Tuxedo Adapter to create adapter services, you must:

- Ensure that each Integration Server in the cluster contains an identical set of packages (see "Replicating Packages to webMethods Integration Servers" on page 29).
- Disable the redirection capability for certain predefined administrative services (see "Disabling the Redirection of Administrative Services" on page 29).

Replicating Packages to webMethods Integration Servers

Every Integration Server in the cluster should contain an identical set of packages that you define using the Tuxedo Adapter. That is, you should replicate the Tuxedo Adapter services and the connections they use.

To ensure consistency, we recommend that you create all packages on one server, and replicate them to the other servers. If you allow different servers to contain different services, you might not derive the full benefits of clustering. For example, if a client requests a service that resides in only one server, and that server is unavailable, the request cannot be successfully redirected to another server.

For information about replicating packages, see the appropriate *webMethods Integration Server Administrator's Guide* for your release.

Disabling the Redirection of Administrative Services

A server that cannot handle a client's service request can automatically redirect the request to another server in the cluster. However, the Tuxedo Adapter uses certain predefined administrative services that you should not allow to be redirected. These services are used internally when you

configure the adapter. If you allow these services to be redirected, your configuration specifications might be saved on multiple servers, which is an undesirable result. For example, if you create two Tuxedo Adapter services, one might be stored on one server, while the other one might be stored on another server. Remember that all adapter services must reside on all webMethods Integration Servers in the cluster.

To disable the redirection of administrative services

1. Shut down the Integration Server Administrator. For the procedure to do this, see the *webMethods Integration Server Administrator's Guide* for your release.

2. Open the following file:

   Integration Server_directory \config\redir.cnf

3. Add the following line to the file:

   ```
   <value name="wm.art">false</value>
   ```

4. Save the file and restart the Integration Server.

Clustering Considerations and Requirements

Note:
The following sections assume that you already have configured the Integration Server cluster. For details about Integration Server clustering, see the *webMethods Integration Server Clustering Guide* for your release.

The following considerations and requirements apply to the Tuxedo Adapter in a clustered environment.

Requirements for Each Integration Server in a Cluster

The following table describes the requirements of each Integration Server in a given cluster:

All Integration Servers in a given cluster must have identical...	For Example...
Integration Server versions	One Integration Server in the cluster cannot be version 7.1.1 and another Integration Server in the cluster be version 6.5.
Adapter packages	All Tuxedo Adapter packages in the cluster must have the same version number, with the same service packs and fixes.
Adapter versions	You should have the same patch level for the adapter on all Integration Servers in the cluster. (For WmTuxedoAdapter 6.0, the patch level is 3.0.)

All Integration Servers in a given cluster must have identical...	For Example...
Adapter connections	If you configure a connection to the Jolt server, this connection must appear on all servers in the cluster so that any Integration Server in the cluster can handle a given request identically. If you plan to use connection pools in a clustered environment, see "Considerations When Configuring Connections with Connection Pooling Enabled" on page 31.
Adapter services	If you configure a specific Tuxedo Adapter service, this same adapter service must appear on all servers in the cluster so that any Integration Server in the cluster can handle the request identically. If you allow different Integration Servers to contain different services, you might not derive the full benefits of clustering. For example, if a client requests a service that resides on only one server, and that server is unavailable, the request cannot be successfully redirected to another server.

For information about replicating adapter packages, connections, and adapter services across multiple Integration Servers in a cluster, see "Replicating Packages to webMethods Integration Servers" on page 29.

Considerations When Installing Tuxedo Adapter Packages

For each Integration Server in the cluster, use the standard Tuxedo Adapter installation procedures, as described in "Installing and Uninstalling the Tuxedo Adapter " on page 21.

Considerations When Configuring Connections with Connection Pooling Enabled

When you configure a connection that uses connection pools in a clustered environment, be sure that you do not exceed the total number of connections that can be opened simultaneously.

For example, if you have a cluster of two Integration Servers with a connection configured to a Jolt server that supports a maximum of 100 connections, the total number of connections possible at one time must not exceed 100. This means that you cannot configure a connection with an initial pool size of 100 and then replicate the connection to both servers because a total of 200 connections could then be opened on the Jolt server.

In another example, consider a connection configured with an initial pool size of 10 and a maximum pool size of 100. If you replicate this connection across a cluster with two Integration Servers, it is possible for the connection pool size on both servers to exceed the maximum number of connections that can be open at one time.

For information about configuring connections for the Tuxedo Adapter, see "Adapter Connections" on page 33.

For more general information about connection pools, see the *webMethods Integration Server Administrator's Guide* for your release.

4 Adapter Connections

Overview

This chapter describes how to configure and manage Tuxedo Adapter connections. For more information about how adapter connections work, see "Adapter Connections" on page 33.

Note:
You must have Integration Server administrator privileges to access the Tuxedo Adapter administrative screens. For information about setting user privileges, see the *webMethods Integration Server Administrator's Guide* for your release.

Before Creating Adapter Connections

Perform the following tasks before creating adapter connections.

To prepare for creating a Tuxedo Adapter connection

1. Install the webMethods Integration Server and the Tuxedo Adapter on the same machine. For instructions, see "Installing and Uninstalling the Tuxedo Adapter " on page 21.

2. Make sure you have Integration Server administrator privileges so that you can access the Tuxedo Adapter administrative screens. For information about setting user privileges, see the *webMethods Integration Server Administrator's Guide* for your release.

3. Start your Integration Server and the Integration Server Administrator, if they are not already running.

4. Using the Integration Server Administrator, make sure the WmTuxedoAdapter package is enabled. For instructions, see "Enabling Packages" on page 27.

5. Using Digital Event Services or Software AG Designer, create a user-defined package to contain the connection, if you have not already done so. For information about how to create a package, see the *webMethods Service Development Help* for your release.

6. Create your connections, as described in "Creating Adapter Connections" on page 34.

Creating Adapter Connections

When you configure Tuxedo Adapter connections, you specify information that the Integration Server uses to connect to the Jolt server.

You create Tuxedo Adapter connections using the Integration Server Administrator.

To create a Tuxedo Adapter connection

1. Start the Integration Server Administrator if it is not already running.

2. Make sure the WmTuxedoAdapter package is enabled. See "Enabling Packages" on page 27 for details.

3. In the Integration Server Administrator, in the **Adapters** menu, click **Tuxedo Adapter**.

4. On the **Connections** screen, click **Configure New Connection**.

5. On the **Connection Types** screen, click **Tuxedo Sync Connection** to display the Configure Connection Type screen.

6. In the **Tuxedo Adapter**section, provide values for the following fields.

Field	Description/Action
Package	The package in which to create the connection. You must create the package using Digital Event Services or Designer before you can specify it using this parameter. For general information about creating packages, see the *webMethods Service Development Help* for your release. **Note:** Create the connection in a user-defined package rather than in the adapter package. For other important considerations when creating packages for the Tuxedo Adapter, see "Managing the Adapter Package" on page 26.
Folder Name	The folder in which to create the connection.
Connection Name	The name you want to give the connection. Connection names cannot have spaces or use special characters reserved by the Integration Server, Digital Event Services, or Designer. For more information about the use of special characters in package, folder, and element names, see the *webMethods Service Development Help* for your release.

7. In the **Connection Properties** section, provides values for the following fields:

Field	Description
Host Server Name	The host name or IP address of the machine on which the Jolt server is running.
Host Port Number	The port number where the Jolt server listens for client requests.
Jolt Transaction Timeout	Optional. The number of seconds that the Tuxedo Adapter will wait during a transaction with the Jolt server before it times out and returns an error. Default: 30 seconds.
User Name	Optional. The user name that you use to log into the Jolt server.

Field	Description
User Role	Optional. The type of user access to the Jolt server. Default: joltadmin.
User Password	Optional. The password associated with the **User Name**.
Retype User Password	Optional. The same user password in the previous field.
Application Name	Optional. The name of the Tuxedo application that you are accessing with this connection.
Application Password	Optional. The password to the Tuxedo application.
Retype Application Password	Optional. The same application password in the previous field.

8. In the **Connection Management Properties** section, use the following fields:

Parameter	Description/Action
Enable Connection Pooling	Enables the connection to use connection pooling. See "Adapter Connections" on page 33 for more information about connection pooling. **Note:** If you plan to enable connection pooling in a clustered environment, consider the connection pool size. For details, see "Considerations When Configuring Connections with Connection Pooling Enabled" on page 31.
Minimum Pool Size	The minimum number of connection objects that remain in the connection pool at all times. When the adapter creates the pool, it creates this number of connections. Default: **1**.
Maximum Pool Size	The maximum number of connection objects that can exist in the connection pool. When the connection pool has reached its maximum number of connections, the adapter will reuse any inactive connections in the pool or, if all connections are active, it will wait for a connection to become available. Default: **10**.
Pool Size Increment	If connection pooling is enabled, this field specifies the number of connections by which the pool will be incremented if connections are needed, up to the maximum pool size. Default: **1**.
Block Timeout (msec)	If connection pooling is enabled, this field specifies the number of milliseconds that the Integration Server will wait to obtain a connection

Parameter	Description/Action
	with the Jolt Server before it times out and returns an error. Default: **1000**.
Expire Timeout (msec)	If connection pooling is enabled, this field specifies the number of milliseconds that an inactive connection can remain in the pool before it is closed and removed from the pool. For example, to specify 10 seconds, specify `10000`. Enter `0` to specify no timeout. Default: **1000**. **Note:** Note that the adapter will never violate the **Minimum Connections** parameter. These connections remain in the pool regardless of how long they are inactive.
Startup Retry Count	If connection pooling is enabled, this parameter specifies the number of times that the system should attempt to initialize the connection pool at startup if the initial attempt fails, before issuing an AdapterConnectionException. Default: 0.
Startup Backoff Timeout	If connection pooling is enabled, this parameter specifies the number of seconds to wait between each attempt to initialize the connection pool. Default: 10.

9. Click **Save Connection**.

10. Enable the connection before you create adapter services that use it. For instructions, see "Enabling Adapter Connections" on page 41.

    The connection you created appears on the adapter's Connections screen and in the Digital Event Services Service Browser.

Dynamically Changing Adapter Connections at Run Time

You can run an adapter service using a connection other than the default connection that was associated with the service when the service was created. To override the default, you must pass a value through the pipeline into the *$connectionName* variable.

For example, you have a flow that creates an account on the production application server. However, you also want the flow to have the capability to create the account on the test server at run time. To facilitate this capability, you can create a branch step in the flow logic. The branch step allows you to conditionally execute a step based on the value of a variable at run time. You can design your flow to pass this value to *$connectionName*based on the conditions you set. For more information about building flow services and using the branch step, see the *webMethods Service Development Help* for your release.

Keep in mind these restrictions when using dynamic connections:

- The Tuxedo services invoked by the adapter service must be deployed on both application servers.

- The *$connectionName* variable is present only in services:
  - Created with Digital Event Services 6.5 or later
  - Created with Digital Event Services 6.1 (or earlier) and edited with Digital Event Services 6.5 or later
  - Created with Designer

For more information, see Changing the Connection Associated with an Adapter Service at Design Time.

Viewing Adapter Connection Parameters

You can view a connection's parameters from the Integration Server Administrator, Digital Event Services, or Designer.

Viewing Adapter Connection Parameters with Integration Server Administrator

Perform the following procedure to use Integration Server Administrator to view adapter connections.

To view the parameters for a Tuxedo Adapter connection using the Integration Server Administrator

1. Start the Integration Server Administrator if it is not already running.

   When using the adapter with Integration Server 8.0 and later, you can sort and filter the list of connections that appears on the Connections screen.

   - To sort information on the Connections screen, click the **Up** and **Down** arrows at the top of the column you want to sort.
   - To filter the list of connections:
     1. On the Connections screen, click **Filter Connections**.
     2. Type the criterion by which you want to filter into the **Filter criteria** box. Filtering is based on the node name, not the connection alias. To locate all connections containing specific alphanumeric characters, use asterisks (*) as wildcards. For example, if you want to display all connections containing the string "abc", type `*abc*` in the **Filter criteria** box.
     3. Click **Submit**. The Connections screen displays the connections that match the filter criteria.
     4. To re-display all connections, click **Show All Connections**.

The Connections screen appears, listing all the current connections. You can control the number of connections that are displayed on this screen. For more information, see the *webMethods Integration Server Administrator's Guide* for your release.

2. Make sure the WmTuxedoAdapter package is enabled. For instructions about enabling packages, see "Enabling Packages" on page 27.

3. In the **Adapters** menu in the navigation area of the Integration Server Administrator, click **Tuxedo Adapter**.

4. On the **Connections** screen, click the icon for the connection you want to see.

   The **View Connection** screen displays the parameters for the connection. For descriptions of the connection parameters, see the table of parameters in "Creating Adapter Connections" on page 34.

5. Click **Return to Tuxedo Adapter Connections** to return to the main connections screen.

Viewing Adapter Connection Parameters With Digital Event Services or Designer

Perform the following procedure to use Digital Event Services or Designer to view adapter connections.

To view the parameters for a Tuxedo Adapter connection using Digital Event Services or Designer

1. Using the Integration Server Administrator, make sure the WmTuxedoAdapter package is enabled. For instructions about enabling packages, see "Enabling Packages" on page 27.

2. From the Digital Event Services or Designer navigation area, open the package and folder in which the connection is located.

3. Double-click the connection you want to view.

   The parameters for the connection appear in the **Connection Information** tab. For descriptions of the connection parameters, see the table of parameters in "Creating Adapter Connections" on page 34.

Editing Adapter Connections

If you want to redefine parameters that a connection uses when connecting to a Jolt server, you can update a connection's parameters using the Integration Server Administrator.

You must disable a connection before you can edit it. For details, see "Disabling Adapter Connections" on page 42.

To edit a Tuxedo Adapter connection

1. Start the Integration Server Administrator if it is not already running.
2. Make sure the WmTuxedoAdapter package is enabled. See "Enabling Packages" on page 27 for details.
3. In the **Adapters** menu in the navigation area of the Integration Server Administrator, click **Tuxedo Adapter**.
4. On the **Connections** screen, click the icon for the connection you want to edit.

   The **Edit Connection** screen displays the current parameters for the connection. Update the connection parameters by typing or selecting the values you want to specify.

   For descriptions of the connection parameters, see the table of parameters in "Creating Adapter Connections" on page 34.
5. Click **Save Changes** to save the connection and return to the **Connections** screen.

Copying Adapter Connections

You can copy an existing Tuxedo Adapter connection to create a new connection with the same or similar connection properties without the need to type all properties for the new connection.

To copy a Tuxedo Adapter connection

1. Start the Integration Server Administrator if it is not already running.
2. Make sure the WmTuxedoAdapter package is enabled. For instructions about enabling packages, see "Enabling Packages" on page 27.
3. In the **Adapters** menu in the navigation area of the Integration Server Administrator, click **Tuxedo Adapter**.
4. On the **Connections** screen, click the **Copy** icon for the connection you want to copy.

   The **Copy Connection** screen displays the current parameters for the connection you want to copy. Name the new connection and edit any connection parameters as needed by typing or selecting the values you want to specify.

   For descriptions of the connection parameters, see the table of parameters in "Creating Adapter Connections" on page 34.
5. Click **Save Connection** to save the connection and return to the **Connections** screen.

Note:

You must enter and then retype a password before you can save the new connection.

6. Enable the connection. For instructions, see "Enabling Adapter Connections" on page 41.

Deleting Adapter Connections

If you no longer want to use a Tuxedo Adapter connection, use the following instructions to delete the connection. You delete adapter connections using the Integration Server Administrator.

If you delete a Tuxedo Adapter connection, the adapter services that use the connection will no longer work. However, if you delete a Tuxedo Adapter connection, you can assign a different connection to an adapter service and re-use the service. To do this, use the pub.art.service:setAdapterServiceNodeConnection service. For more information, see "Changing the Connection Associated with an Adapter Service at Design Time" on page 15.

To delete a Tuxedo Adapter connection

1. Start the Integration Server Administrator.

2. Make sure the WmTuxedoAdapter package is enabled. See "Enabling Packages" on page 27 for details.

3. In the **Adapters** menu in the navigation area of the Integration Server Administrator, click **Tuxedo Adapter**.

4. On the Connections screen, click ✕ for the connection you want to delete.

   The Integration Server deletes the adapter connection.

Enabling Adapter Connections

Tuxedo Adapter connections must be enabled before you can create adapter services for those connections. When you create a connection, it is automatically disabled. The Integration Server only enables the connection if the connection parameters are valid.

Note:
When you reload a package that contains enabled connections, the connections will be enabled automatically when the package reloads. If the package contains connections that are disabled, they will remain disabled when the package reloads.

To enable a Tuxedo Adapter connection

1. Start the Integration Server Administrator if it is not already running.

2. Make sure the WmTuxedoAdapter package is enabled. See "Enabling Packages" on page 27 for details.

3. In the **Adapters** menu in the navigation area of the Integration Server Administrator, click **Tuxedo Adapter**.

4. On the **Connections** screen, click **No** in the **Enabled** column for the connection you want to enable.

   The Integration Server Administrator enables the adapter connection and displays ✔ and **Yes** in the **Enabled** column.

Disabling Adapter Connections

Tuxedo Adapter connections must be disabled before you can edit or delete the connections.

To disable a Tuxedo Adapter connection

1. Start the Integration Server Administrator if it is not already running.

2. Make sure the WmTuxedoAdapter package is enabled. See "Enabling Packages" on page 27 for details.

3. In the **Adapters** menu in the navigation area of the Integration Server Administrator, click **Tuxedo Adapter**.

4. On the **Connections** screen, click **Yes** in the **Enabled** column for the connection you want to disable.

   The adapter connection becomes disabled and you will no longer see a ✔ and **Yes** in the **Enabled** column.

5 Adapter Services

Overview

The following sections describe how to configure adapter services, which you use for Integration Server to Jolt server communications.

Before you configure the Tuxedo Adapter services, you must configure the connections you plan to use with them. For details, see "Creating Adapter Connections" on page 34.

Before Configuring or Managing Adapter Services

Perform the following tasks before configuring or managing adapter services.

To prepare to configure or manage a Tuxedo Adapter service

1. Start your Integration Server and the Integration Server Administrator, if they are not already running.

2. Make sure you have Integration Server administrator privileges so that you can access the Tuxedo Adapter administrative screens. For information about setting user privileges, see the *webMethods Integration Server Administrator's Guide* for your release.

3. Using the Integration Server Administrator, make sure the WmTuxedoAdapter package is enabled. For instructions, see "Enabling Packages" on page 27.

4. Using the Integration Server Administrator, configure an adapter connection to use with the adapter service. For instructions, see "Enabling Adapter Connections" on page 41.

5. Start the Digital Event Services or Software AG Designer if it is not already running.

   Note:
   If you are using Digital Event Services 6.1 or later, use the Edit perspective for all procedures unless stated otherwise. If you are using Designer, use the Service Development perspective. For more information, see the *webMethods Service Development Help* for your release.

6. Using Digital Event Services or Designer, create a package to contain the service, if you have not already done so. When you configure adapter services, define them in user-defined packages instead of in the WmTuxedoAdapter package. For more information about managing packages for the adapter, see "Adapter Package Management" on page 25.

Configuring Synchronous Services

The Tuxedo Adapter provides one service template named Tuxedo Synchronous Service. A Tuxedo Synchronous Service adapter service executes services from a Tuxedo application. For more information about adapter services, see "Using Adapter Services" on page 17.

Note:

Before you configure the adapter service, you must create the adapter connection you plan to use with it. For instructions, see "Adapter Connections" on page 33.

To configure a Tuxedo Synchronous Service

1. Review the steps in "Before Configuring or Managing Adapter Services" on page 44.
2. Start Digital Event Services or Designer.
3. Perform one of the following:
   - If you are using Digital Event Services, select **File > New > Adapter Service**and click **Next**.
   - If you are using Designer, perform the following:
     1. Right-click the package in which the service should be contained and select **New > Adapter Service**.
     2. Select the parent namespace and type a name for the adapter service.
     3. Click **Next**.
4. Select **Tuxedo Adapter** as the adapter type and click **Next**.
5. Select the appropriate **Adapter Connection Name** and click **Next**.
6. Select the**Tuxedo Synchronous Service** template and do one of the following:
   - If you are using Digital Event Services, click **Next**, select a package and folder to contain the service, type a unique name for the service, and click **Finish**.
   - If you are using Designer, click **Finish**.
7. In Digital Event Services or Designer, select and open the service. You can select the **Tuxedo Services**, **Adapter Settings**, or **Input/Output** tabs at any time to view the information.
8. Select the **Tuxedo Services** tab and from the **Select Tuxedo Service** drop-down list box, select a Tuxedo service. The **Tuxedo Services**tab lists the input and output fields, the data types, and the maximum number of occurrences for each field. For information about a specific Tuxedo service, contact your Tuxedo developer.

   Important: All variables in the input signature will be rendered as java.lang.Strings and will appear as strings on the input pipeline regardless of their data type in Tuxedo. The execute() function of the Tuxedo Adapter will translate these strings to the proper input data types before passing them to Tuxedo.
9. From the **File** menu, select **Save**.

Testing Adapter Services

You use Digital Event Services or Designer to test adapter services.

For more information about testing and debugging services, see the *webMethods Service Development Help* for your release.

To test adapter services

1. Review the steps in "Before Configuring or Managing Adapter Services" on page 44.
2. If you are using Digital Event Services, set the view to the Test perspective.
3. In Digital Event Services or Designer, expand the package and folder that contain the service you want to test.
4. Double-click the service you want to test.

   Digital Event Services or Designer displays the configured service in the service template's Adapter Service Editor.
5. Perform one of the following
   - If you are using Digital Event Services, select **Test > Run**.
   - If you are using Designer, select **Run > Run As > Run Service**.
6. For every service input field, you will be prompted to enter an input value. Enter a value for each input field and then click **OK**.
7. Click the **Results** tab (in Digital Event Services) or **Service Result** tab (in Designer) to view the output from this service.

Viewing Adapter Services

You use Digital Event Services or Designer to view adapter services.

To view a service

1. Review the steps in "Before Configuring or Managing Adapter Services" on page 44.
2. In Digital Event Services or Designer, expand the package and folder that contain the service you want to view.
3. Double-click the service you want to view.

Digital Event Services or Designer displays the configured service in the service template's Adapter Service Editor.

Editing Adapter Services

You use Designer to edit adapter services.

To edit an adapter service

1. In Designer, browse to and open the adapter service that you want to edit.
2. Double-click the service that you want to edit.

   Designer displays the adapter service in the service template's Adapter Service Editor.
3. Do one of the following:
   - If you have the VCS Integration feature enabled, right-click the service and select **Check Out**.
   - If you do not have the VCS Integration feature enabled, right-click the service and select **Lock for Edit**.
   - If you are using the local service development feature, from the **Team** menu in Designer, select the appropriate option to check out the service. The options available in the **Team** menu depend on the VCS client that you use.
4. Modify the values for the adapter service's parameters as needed. For detailed descriptions of the service's parameters, see the section on configuring a service for the specific type of service you want to edit.
5. After you complete your modifications, save the service and do one of the following:
   - If you have the VCS Integration feature enabled, right-click the service and select **Check In**. Enter a check-in comment and click **OK**.
   - If you do not have the VCS Integration feature enabled, right-click the service and select **Unlock**.
   - If you are using the local service development feature, from the **Team** menu in Designer, select the appropriate option to check in the service. The options available in the **Team** menu depend on the VCS client that you use.
6. Save the service.

Deleting Adapter Services

You use the Digital Event Services or Designer to delete adapter services.

To delete a service

1. Review the steps in "Before Configuring or Managing Adapter Services" on page 44.
2. In Digital Event Services or Designer, expand the package and folder that contain the service you want to delete.
3. Right-click the adapter service and click **Delete**.

Validating Adapter Service Values

Digital Event Services or Designer enables the Tuxedo Adapter to validate user-defined data for adapter services at design time. You can validate the values for a single adapter service or you can configure Digital Event Services or Designer to always validate the values for adapter services. Both options could potentially slow your design-time operations.

For more information about the **Adapter Service/Notification Editor** and other Digital Event Services or Designer menu options and toolbar icons, see the *webMethods Service Development Help* for your release.

Validating Values for a Single Adapter Service

When you enable data validation for a single adapter service, Digital Event Services or Designer compares the service values against the resource data that has already been fetched from the selected adapter.

To enable automatic data validation for a single adapter service

1. Review the steps in "Before Configuring or Managing Adapter Services" on page 44.
2. In Digital Event Services or Designer, expand the package and folder that contain the service for which you want to enable automatic validation.
3. Double-click the service for which you want to validate the data.

   Digital Event Services or Designer displays the configured adapter service in the service template's Adapter Service Editor.
4. Click the icon.

Validating Values for All Adapter Services

If you select the option to always validate values for adapter services, it will do so for all Integration Server WmART-based adapters installed on the Integration Server.

To always validate the values for all adapter services

1. Review the steps in "Before Configuring or Managing Adapter Services" on page 44.
2. Start Digital Event Services or Designer.
3. Perform one of the following:
   - If you are using Digital Event Services, select the **Tools > Options > Integration Server > Adapter Service/Notification Editor** item.
   - If you are using Designer, select the **Window > Preferences > Software AG > Service Development > Adapter Service/Notification Editor** item.
4. Enable the **Automatic data validation** option.
5. Click **OK**.

Reloading Adapter Values

You can enable the Tuxedo Adapter to reload and validate user-defined data for adapter services at design time in Digital Event Services or Designer. You can reload values for a single adapter service or you can configure Digital Event Services or Designer so it automatically reloads the values for adapter services. Both options could potentially slow your design-time operations.

For more information about the **Adapter Service/Notification Editor**, other menu options, and toolbar icons, see the *webMethods Service Development Help* for your release.

Reloading Values for a Single Adapter Service

When you reload adapter values for a single adapter service, Digital Event Services or Designer compares the service values against the resource data that has already been fetched from the selected adapter.

To reload the adapter values for a single adapter service

1. Review the steps in "Before Configuring or Managing Adapter Services" on page 44.
2. In Digital Event Services or Designer, expand the package and folder that contain the service for which you want to enable automatic validation.
3. Double-click the service for which you want to validate the data.

   Digital Event Services or Designer displays the configured adapter service in the service template's Adapter Service Editor.

4. Click the icon.

Reloading Values for All Adapter Services

If you select the option to always reload values for adapter services, it will do so for all Integration Server WmART-based adapters installed on the Integration Server.

To reload the adapter values for all adapter services

1. Review the steps in "Before Configuring or Managing Adapter Services" on page 44.
2. Start Digital Event Services or Designer.
3. Perform one of the following:
   - If you are using Digital Event Services, select the **Tools > Options > Integration Server > Adapter Service/Notification Editor** item.
   - If you are using Designer, select the **Window > Preferences > Software AG > Service Development > Adapter Service/Notification Editor** item.
4. Enable the **Automatic polling of adapter metadata** option.
5. Click **OK**.

6 Adapter Logging and Exception Handling

Overview

The following sections describe message logging and Tuxedo Adapter exception handling. A list of error codes and supporting information appears at the end of this chapter.

Adapter Logging Levels

The Tuxedo Adapter uses the Integration Server logging mechanism to log messages. You can configure and view the Integration Server logs to monitor and troubleshoot the Tuxedo Adapter. For detailed information about logging in the Integration Server, including instructions for configuring and viewing the different kinds of logs supported by the server, see the *webMethods Audit Logging Guide* for your release.

Configuring Adapter Logging Levels

Beginning with Integration Server 7.1.1, you can configure different logging levels for the Tuxedo Adapter. For complete information about specifying the amount and type of information to include in the log, see the *webMethods Audit Logging Guide* for your release.

Accessing Logging Information

Perform the following to access an adapter's logging information.

To access the adapter's logging information

1. From the Integration Server Administrator screen, select **Settings > Logging**.

   The **Logging Settings** screen appears. The **Loggers** section has **Adapters** included in the **Facility** section.

2. Expand the **Adapters** tree to see a list of all installed adapters with their code number and adapter description, along with the logging level.

Changing Logging Settings

Perform the following to change an adapter's logging settings.

To change logging settings for the adapter

1. Click **Edit Logging Settings**. Select the required **Level of Logging** for the Tuxedo Adapter.

2. After making your changes, click **Save Changes**.

Adapter Message Logging

The Integration Server maintains several types of logs; however, the Tuxedo Adapter logs messages only to the audit, error, and server logs. Because the Tuxedo Adapter works in conjunction with the WmART package, the adapter's messages and exceptions typically appear within log messages for the WmART package.

The logging levels are different depending on which version of the Integration Server you are running the adapter on, as shown in the following table.

Integration Server	Log	Description
Integration Server 6.5	Audit Log	You can monitor individual adapter services using the audit log as you would audit any service in the Integration Server. The audit properties for an adapter service are available in the Tuxedo Adapter service template on the **Audit** tab.
	Error Log	The Tuxedo Adapter automatically posts critical-level and error-level log messages to the error log. These log messages appear as adapter run-time messages.
	Server Log	The Tuxedo Adapter posts messages to the server log, depending on how the server log is configured. Critical-level through debug-level log messages appear as adapter run-time log messages. V1-Verbose1 or V4-Verbose4 log messages appear as Tuxedo Adapter log messages.
Integration Server 7.1.1 or higher	Audit Log	You can monitor individual adapter services using the audit log as you would audit any service in the Integration Server. The audit properties for an adapter service are available in the Tuxedo Adapter service template on the **Audit** tab.
	Error Log	The Tuxedo Adapter automatically posts fatal-level and error-level log messages to the error log. These log messages appear as adapter run-time messages.
	Server Log	The Tuxedo Adapter posts messages to the server log, depending on how the server log is configured. Fatal-level through debug-level log messages appear as adapter run-time log messages. Trace-level log messages appear as Tuxedo Adapter log messages.

The Tuxedo Adapter's log messages appear in the format, ADA.740.*nnnnc*, where:

- ADA is the facility code that indicates the message is from an adapter.
- 740 is the Tuxedo Adapter major error code, which indicates that the message is generated by the Tuxedo Adapter.
- *nnnn* represents the error's minor code. For detailed descriptions of the Tuxedo Adapter's minor codes, see "Adapter Error Codes" on page 55.
- *c* represents the message's severity level (optional).

To monitor the Tuxedo Adapter's log messages in the server log, ensure that your server log's logging settings are configured to monitor the following facilities:

- 0113 Adapter Runtime (Managed Object)
- 0114 Adapter Runtime
- 0117 Adapter Runtime (Adapter Service)
- 0118 Adapter Runtime (Connection)
- 0121 Adapter Runtime (SCC Transaction Manager)
- 0126 Adapter Runtime (SCC Connection Manager)

Adapter Exception Handling

The Tuxedo Adapter throws two kinds of exceptions that you should be aware of as you build integrations using the adapter:

- AdapterException
- AdapterConnectionException

When creating a flow or Java service that incorporates an adapter service, you might want to build logic into the wrapping service to catch and handle these types of exceptions.

AdapterException

The Tuxedo Adapter throws an AdapterException to report an error related to the adapter's logic, such as a configuration error or a connection creation error.

To manage the AdapterException, you can catch the DetailedServiceException in a flow or Java service and then navigate through the nested exceptions to the AdapterException, which will contain the error code identifying the error.

AdapterConnectionException

The Tuxedo Adapter throws an AdapterConnectionException to wrap a Jolt exception if the adapter interprets the code as a fatal error.

In this case, WmART drops the connection from the connection pool and tries to create a new connection. It then wraps the exception in com.wm.pkg.art.error.DetailedSystemException and throws it to the Integration Server.

Customizing the Adapter's List of Fatal Error Codes

The execution of a Tuxedo Adapter service may sometimes result in a bea.jolt.JoltException being thrown. All the Jolt exceptions are distinguished by an error number. The list of these Jolt exception error numbers can be found on the web site of Oracle.

By default, a Jolt exception with the error number TPEJOLT is considered as a fatal exception, which results in an AdapterConnectionException being thrown. You can also add a specific Jolt error number to the list of fatal Jolt error numbers.

If a Jolt exception error number matches TPEJOLT or any of the error numbers in the fatal error number list, then the adapter throws an AdapterConnectionException. The Adapter Runtime then catches the AdapterConnectionException, and automatically refreshes the corresponding adapter connections.

If the Jolt exception error number does not match TPEJOLT or any of the error numbers in the fatal error number list, then the adapter throws an AdapterException.

To add JoltException error numbers to the list of fatal error numbers

1. Start the Integration Server Administrator if it is not already running.
2. Under **Settings** in the left panel, select **Extended**.
3. Select **Edit Extended Settings**. In the edit box, type:

   ```
   watt.tuxedoadapter.fatalErrors=+ErrorNumber_1, ErrorNumber_2, ErrorNumber_n
   ```

   Example: To allow the Tuxedo Adapter to refresh connections when encountering JoltException with error numbers TPESYSTEM, TPEIO, and TPEOS, type:

   ```
   watt.tuxedoadapter.fatalErrors=+TPESYSTEM, TPEIO, TPEOS
   ```

4. Click **Save Changes**.
5. Restart the Tuxedo Adapter.

Adapter Error Codes

The Tuxedo Adapter categorizes its minor code numbers as follows:

Error Code	Description
3000-3016	Connection errors between the adapter and the Jolt server.

Error Code	Description
302*x*	Errors from a failed local transaction. These can occur at design time or at run time.
5000-5011	Cache look ups in the connection factory errors. These can occur at design time or at run time.
501*x*-502*x*	Errors from the introspection of the Jolt server's repository. This repository contains the names and input/output signatures of all remote Tuxedo services as well as all the associated data type information. During the introspection sequence, a connection is established, a list of all service names and their input/output metadata parameters is captured and placed in a cache in the connection factory object of the connection for access during adapter service creation (design time) or during adapter service execution (run time). 501*x* - Input Signature Creation Errors 502*x* - Output Signature Creation Errors
503*x*	Errors from executing the adapter services.

The following lists the Tuxedo Adapter's minor codes and provides information about the error message, reason, and possible action for each error.

Error Code	Description
3001	**Can't instantiate JOLT Parameter Object. Check Jolt.jar file.**
	Explanation: The jolt.jar file is missing.
	Action: Verify that the jolt.jar file exists in the WmTuxedoAdapter/code/jars folder. If it does not exist, copy the jolt.jar file from the *Jolt_directory*\udataobj\jolt\java\lib directory where *Jolt_directory* is the directory where the Jolt server is running.
3002	**Failed to get Tuxedo6SyncConnectionFactory object from parent.**
	Explanation: The Tuxedo connection cannot retrieve a reference to the connection factory.
	Action: Check the error log for connection errors and reload/restart the adapter package.
3003	**TuxedoSyncConnection:JoltConnection open() error.**
	Explanation: The Tuxedo connection caught an exception from the Jolt API while attempting to connect.
	Action: Verify that the jolt.jar file exists in the WmTuxedoAdapter/code/jars folder. If it does not exist, copy the jolt.jar file from the

Error Code	Description
	Jolt_directory\udataobj\jolt\java\lib directory where *Jolt_directory* is the directory where the Jolt server is running.
3004	**TuxedoSyncConnection:Failed to open() Jolt Connection.**
	Explanation: The Tuxedo connection failed to successfully open a Jolt server connection.
	Action: Check that the connection parameters (hostname, port, login, etc.) are correct for the connection, and verify that the physical connection still exists.
3005	**TuxedoSyncConnection:registerResourceDomain() failed.**
	Explanation: The Tuxedo connection cannot register the adapter service templates for this connection.
	Action: Check the error log for connection errors and reload/restart the adapter package.
3010	**TuxedoSyncConnection:Failed to get JoltSession object: SessionException received.**
	Explanation: The Tuxedo connection was unable to establish a session with the Jolt server.
	Action: Verify that the jolt.jar file exists in the WmTuxedoAdapter/code/jars folder.If it does not exist, copy the jolt.jar file from the *Jolt_directory*\udataobj\jolt\java\lib directory where *Jolt_directory* is the directory where the Jolt server is running. Check that the connection parameters (hostname, port, login, etc) are correct for the connection, and verify that the physical connection still exists.
3011	**TuxedoSyncConnection:Failed to close a JoltSession in endSession().**
	Explanation: The Tuxedo connection was unable to end a session with the Jolt server.
	Action: Check the session timeout definition on the Jolt server. It should be set to infinite.
3012	**TuxedoSyncConnection:Failed to Get/Set JoltSessionAttributes in Connection open().**
	Explanation: The Jolt API is unable to return or set connection attributes.
	Action: Verify that the jolt.jar file exists in the WmTuxedoAdapter/code/jars folder. If it does not exist, copy the jolt.jar file from the *Jolt_directory*\udataobj\jolt\java\lib directory where *Jolt_directory* is the directory where the Jolt server is running. Verify that the parameters for the connection are correct.

Error Code	Description
3013	**TuxedoSyncConnection:AdapterCheckValue() Failed.**
	Explanation: The connection is unable to get the adapter service object for an adapter service name.
	Action: Check the adapter service. It may be deleted, corrupted, or renamed.
3014	**TuxedoSyncConnection:adapterResourceDomainLookup() Failed.**
	Explanation: The connection is unable to get the ResourceDomainValues for an adapter service name.
	Action: Check the adapter service. It may be deleted, corrupted, or renamed.
3015	**TuxedoSyncConnectionFactory:can't get the Repository Object.**
	Explanation: The Tuxedo connection factory cannot create a Jolt repository introspection object.
	Action: Check the error log for details. Verify that the parameters for the connection are correct.
3016	**TuxedoSyncConnectionFactory:initRepository() Failed.**
	Explanation: The Tuxedo connection factory received an exception from the Jolt interface while attempting to create a Jolt Repository introspection object.
	Action: Check the error log for details. Verify that the parameters for the connection are correct.
3020	**TuxedoLocalTransaction:Failed to begin Jolt Transaction.**
	Explanation: The LOCAL TRANSACTION is unable to create a Jolt server transaction object.
	Action: Check the error log for details. Also verify that the Tuxedo connection status is enabled and that the parameters for this connection are correct.
3021	**TuxedoLocalTransaction:Failed to commit Jolt Transaction.**
	Explanation: The LOCAL TRANSACTION is unable to commit a Jolt server transaction object.
	Action: Check the error log for details. Also verify that the Tuxedo connection status is enabled and that the parameters for this connection are correct.
3022	**TuxedoLocalTransaction:Failed to rollback Jolt Transaction.**
	Explanation: The LOCAL TRANSACTION is unable to rollback a Jolt server transaction object.

Error Code	Description
	Action: Check the error log for details. Also verify that the Tuxedo connection status is enabled and that the parameters for this connection are correct.
5001	**TuxedoAdapter:MetadataCommon can't find Service Descriptor, Cache Null.**
	Explanation: Unable to look up a Tuxedo service in the connection factory. The cache pointer is NULL.
	Action: Check the error log for details and make sure the Tuxedo connection is enabled and has no connection failures.
5002	**TuxedoAdapter:MetadataCommon can't find Service Descriptor in Cache.**
	Explanation: Unable to lookup a Tuxedo service in the connection factory. Name is not in the cache.
	Action: Check the error log for details and make sure the Tuxedo connection is enabled and has no connection failures.
5010	**TuxedoAdapter:TuxRepoLookupName can't get factory from connection.**
	Explanation: The Tuxedo connection is unable to get the reference to the parent connection factory.
	Action: Check the error log for details. Reload/restart the Tuxedo Adapter package.
5011	**TuxedoAdapter:TuxRepoLookupName can't get Repository Cache from Factory.**
	Explanation: The Tuxedo connection factory is unable to return the pointer to the repository cache.
	Action: Check the connection status and look in the error log for connection errors.
5012	**TuxedoAdapter:TuxInputSignature can't get factory from connection.**
	Explanation: The Tuxedo connection is unable to get the reference to the parent connection factory.
	Action: Check the error logs for connection errors. Reload/restart the Tuxedo Adapter package.
5013	**TuxedoAdapter:TuxInputSignature can't get Repository Cache from Factory.**
	Explanation: The Tuxedo connection factory is unable to return the pointer to the repository cache.
	Action: Check the connection status and look in the error log for connection errors.
5014	**TuxedoAdapter:TuxInputSignature No Service Descriptors in Repository.**

Error Code	Description
	Explanation: The repository cache in the connection factory is empty.
	Action: Check the connection status and verify that the connection attributes (Application Name, Application Password, User Role, etc.) are correct.
5015	**TuxedoAdapter:TuxInputSignature Service Name is NULL.**
	Explanation: The service name passed for look ups in the repository cache is NULL.
	Action: The adapter service is corrupt, deleted, renamed, or out of sync with the repository.
5016	**TuxedoAdapter:TuxInputSignature can't Find Service Descriptor in Repository.**
	Explanation: The service name passed for look ups in the repository cache cannot be found.
	Action: The cache in the connection factory is not synchronized with the repository. Reload/restart the adapter.
5017	**TuxedoAdapter:TuxInputSignature Input Signature in Descriptor is NULL.**
	Explanation: The Tuxedo service descriptor in the repository cache has no input signature.
	Action: Check the input signature definition on the Jolt server for this service.
5018	**TuxedoAdapter:TuxInputSignature No Parameters in Input Signature Descriptor.**
	Explanation: The input signature in the repository cache has no parameters.
	Action: Check the input signature definition on the Jolt server for this service.
5022	**TuxedoAdapter:TuxOutputSignature can't get factory from connection.**
	Explanation: The Tuxedo connection is unable to get the reference to the parent connection factory.
	Action: Check the error log for details. Reload/restart the Tuxedo Adapter package.
5023	**TuxedoAdapter:TuxOutputSignature can't get Repository Cache from Factory.**
	Explanation: The Tuxedo connection factory is unable to return the pointer to the repository cache.
	Action: Check the connection status and look in the error log for connection errors.
5024	**TuxedoAdapter:TuxOutputSignature No Service Descriptors in Repository.**
	Explanation: The repository cache in the connection factory is empty.

Error Code	Description
	Action: Check the connection status and verify that the connection attributes (Application Name, Application Password, User Role, etc.) are correct.
5025	**TuxedoAdapter:TuxOutputSignature Service Name is NULL.**
	Explanation: The service name passed for look ups in the repository cache is NULL.
	Action: The adapter service is corrupt, deleted, renamed, or not synchronized with the repository.
5026	**TuxedoAdapter:TuxOutputSignature can't find Service Descriptor in Repository.**
	Explanation: The service name passed for look ups in the repository cache cannot be found.
	Action: The cache in the connection factory is not synchronized with the repository. Reload/restart the adapter.
5027	**TuxedoAdapter:TuxOutputSignature Output Signature in Descriptor is NULL.**
	Explanation: The Tuxedo service descriptor in the repository cache has no output signature.
	Action: Check the output signature definition on the Jolt server for this service.
5028	**TuxedoAdapter:TuxOutputSignature No Parameters in Output Signature Descriptor.**
	Explanation: The output signature in the repository cache has no parameters.
	Action: Check the output signature definition on the Jolt server for this service.
5030	**TuxedoAdapter:Tux Synch Service, execute:Input WmRecord is NULL.**
	Explanation: The input signature record passed into the adapter service is NULL.
	Action: Check the error log for details. Reload/restart the adapter package.
5031	**TuxedoAdapter:Tux Synch Service, execute:can't create JoltRemoteService Object.**
	Explanation: Unable to create a Jolt service with the name given in the adapter service. Adapter service is not synchronized with the Jolt server repository.
	Action: Reload or restart the adapter.
5032	**TuxedoAdapter:Tux Synch Service, execute:JoltRemoteService.call() failed.**
	Explanation: The call() to execute the remote service on the Jolt server failed. Check the error log for details.

Error Code	Description
	Action: Check the connection status and the adapter service. Ensure that the adapter service is synchronized with the Tuxedo service.
5033	**TuxedoAdapter:Tux Synch Service, execute:Input Field: can't cast from Java-type to Jolt-Type.**
	Explanation: The input field data type for Java cannot be translated to a Jolt server data type.
	Action: Check the data type definitions in the adapter service in Designer.
5034	**TuxedoAdapter:Tux Synch Service, execute:Input Fields: No Service Input Values.**
	Explanation: There are no input signature values to pass to the Jolt remote service.
	Action: Check the input signature definition in Designer.
5035	**TuxedoAdapter:Tux Synch Service, execute:WmManagedConnection is not Tuxedo6SyncConnection.**
	Explanation: The adapter service template does not match the connection type. Check the error log for details.
	Action: Reload/restart the adapter package.
5036	**TuxedoAdapter:Tux Synch Service, execute:Error while adding Values to RemoteService.**
	Explanation: The adapter service cannot add the input signature values to the Jolt remote service object.
	Action: Compare the adapter service signature to the Jolt server repository signature.
5037	**TuxedoAdapter:Tux Synch Service, execute:Error reading response data from JoltRemoteService.**
	Explanation: Data defined in the adapter service output signature was not returned by the Jolt remote service.
	Action: Compare the adapter service signature to the Jolt server repository signature.
5038	**TuxedoAdapter:Tux Synch Service, execute:can't translate IS datatype for input field to Tuxedo datatype.**
	Explanation: A data type in the adapter service input signature cannot be translated to a Jolt remote service data type.
	Action: Compare the adapter service signature to the Jolt server repository signature.

Error Code	Description
5039	**TuxedoAdapter:Tux Synch Service, execute:can't translate Tux datatype to IS datatype for output field.**
	Explanation: The data returned by the Jolt remote service contains a data type that cannot be translated to a Java data type.
	Action: Compare the adapter service signature to the Jolt server repository signature.

7 Tuxedo Adapter Administrator APIs

Tuxedo Adapter Administrator APIs

The administrator APIs are available for Tuxedo Adapter. For more information, see *webMethods Adapter Runtime User's Guide*.

Sample Template for creating Connections:

```
{
  "connectionAlias": "Connection Alias",
  "adapterTypeName": "com.wm.adapter.tuxedo6adapter.Tuxedo6Adapter",
  "connectionFactoryType":
"com.wm.adapter.tuxedo6adapter.connections.Tuxedo6SyncConnectionFactory",
  "packageName": "Package Name",
  "connectionSettings": {
    "transactionType": "LOCAL_TRANSACTION",
    "hostName": "10.60.27.187",
    "portNumber": "3555",
    "connTimeout": "30",
    "userName": "username",
    "userRole": "Admin",
    "userPassword": "Password",
    "appName": "App",
    "appPassword": "Password"
  },
  "connectionManagerSettings": {
    "poolable": "true",
    "minimumPoolSize": "1",
    "maximumPoolSize": "10",
    "poolIncrementSize": "1",
    "blockingTimeout": "1000",
    "expireTimeout": "1000",
    "startupRetryCount": "0",
    "startupBackoffSecs": "10"
  }
}
```

A Built-In Transaction Management Services

Transaction Management Overview

This appendix provides an overview and examples of using transactions. It describes how the Integration Server supports the built-in services used to manage explicit transactions for your Tuxedo Adapter services in the WmART package. For descriptions of each of the specific built-in transaction management services that can be used with the WmART package, see "Built-In Transaction Management Services" on page 67.

For information about other built-in services available with the Tuxedo Adapter, see the *webMethods Integration Server Built-In Services Reference* for your release.

Transactions

The Integration Server considers a transaction to be one or more interactions with one or more resources that are treated as a single logical unit of work (LUW). The interactions within a transaction are either all committed or all rolled back. For example, if a transaction includes one or more calls to Tuxedo Adapter services and one of these services fails, all other services in the transaction are rolled back.

Transaction Types

The Tuxedo Adapter supports local transactions with the Jolt server. For a description of the transaction types supported by the Tuxedo Adapter, see "Transaction Management of Adapter Connections" on page 15.

In general, the Tuxedo Adapter can communicate with multiple Jolt servers (resources) at a time. There are limitations when the adapter tries to access multiple resources using locally transacted connections within a single flow service. Also, the Jolt server can impose its own set of restrictions on how a connection may be used in a transaction.

Note:
If a transaction accesses multiple resources, and more than one of the resources only supports only local transactions, the integrity of the transaction cannot be guaranteed. For example, if the first resource successfully commits, and the second resource fails to commit, the first resource interaction cannot be rolled back; it has already been committed. To help prevent this problem, the Integration Server detects this case when connecting to more than one resource that does not support two-phase commits. It throws a run-time exception and the service execution fails.

Implicit and Explicit Transactions

Implicit transactions are handled automatically by the Integration Server transaction manager. Implicit transaction support is enabled when you configure an adapter service to use a Tuxedo Adapter Connection and invoke this adapter service from a flow. When you define an explicit transaction, you define the start-on-completion boundaries of the transaction. As such, implicit and explicit transactions need to be created and managed differently.

The following sections describe implicit and explicit transactions and how to manage them.

Implicit Transactions

With implicit transactions, the Integration Server automatically manages local transactions without requiring you to explicitly do anything. That is, the Integration Server starts and completes an implicit transaction with no additional service calls required by the adapter user.

A transaction context, which the transaction manager uses to define a unit of work, starts when a flow service executes an adapter service. The connection required by the adapter service is registered with the newly created context and used by the adapter service. If the flow executes another adapter service, the transaction context is searched to see if the connection is registered already. If the connection is already registered, the adapter service uses this connection. If the connection is not registered, the Integration Server retrieves a new connection instance and registers it with the transaction.

Note that if the top level flow invokes another flow, adapter services in the child flow use the same transaction context.

When the top level flow completes, the transaction completes and either is committed or rolled back, depending on the status (success or failure) of the top level flow.

A single transaction context can contain no more than one Tuxedo Adapter Connection. If your flow contains adapter services that use more than one Tuxedo Adapter Connection, you must use explicit transactions, which are described in the next section.

For more information about designing and using flows, see the *webMethods Service Development Help* for your release.

For more information about transaction types, see "Transaction Management of Adapter Connections" on page 15.

Explicit Transactions

You use explicit transactions when you need to explicitly control the transactional units of work. To do this, you use additional services, known as built-in services, in your flow.

A transaction context starts when the pub.art.transaction.startTransaction() service executes. The transaction context completes when either the pub.art.transaction.commitTransaction() or pub.art.transaction.rollbackTransaction() service executes. As with implicit transactions, a single transaction context can contain no more than one Tuxedo Adapter Connection connection.

Note:
With explicit transactions, you must be sure to call either a commitTransaction() or rollbackTransaction() for each startTransaction() service, or you will have dangling transactions, which will require you to reboot the Integration Server.

A new explicit transaction context can be started within an existing transaction context, provided that you ensure that the transactions are committed in the reverse order they were started-that is, the last transaction to start should be the first transaction to complete, and so forth.

For example, consider the following is a valid construct:

```
pub.art.transaction.startTransaction()
pub.art.transaction.startTransaction()
pub.art.transaction.startTransaction()
pub.art.transaction.commitTransaction()
pub.art.transaction.commitTransaction()
pub.art.transaction.commitTransaction()
```

The following example shows an invalid construct:

```
pub.art.transaction.startTransaction()
pub.art.transaction.startTransaction()
pub.art.transaction.commitTransaction()
pub.art.transaction.commitTransaction()
```

For more information about designing and using flows, see the *webMethods Service Development Help* for your release.

For more information about transaction types, see "Transaction Management of Adapter Connections" on page 15.

Built-In Transaction Management Services

The following sections describe each of the built-in services you can use with the WmART package.

pub.art.transaction:commitTransaction

This service commits an explicit transaction. It must be used in conjunction with the pub.art.transaction:startTransaction service. If it does not have a corresponding pub.art.transaction:startTransaction service, your flow service will receive a runtime error.

For more information about implicit and explicit transactions, see "Transaction Management Overview" on page 68.

Input Parameters

commitTransactionInput	**Document.** A document that contains the variable *transactionName*, described below.
transactionName	**String.** Used to associate a name with an explicit transaction. The *transactionName* must correspond to the *transactionName* in any pub.art.transaction:startTransaction or pub.art.transaction:rollbackTransaction services associated with the explicit transaction. This value must be mapped from the most recent pub.art.transaction:startTransaction that has not previously been committed or rolled back.

Output Parameters

None.

pub.art.transaction:rollbackTransaction

This service rolls back an explicit transaction. It must be used in conjunction with the pub.art.transaction:startTransaction service. If it does not have a corresponding pub.art.transaction:startTransaction service, your flow service will receive a runtime error.

For more information about implicit and explicit transactions, see "Transaction Management Overview" on page 68.

Input Parameters

rollbackTransactionInput	**Document.** A document that contains the variable *transactionName*, described below.
transactionName	**String.** Used to associate a name with an explicit transaction. The *transactionName* must correspond to the *transactionName* in any WmART.pub.art.transaction:startTransaction or WmART.pub.art.transaction:commitTransaction services associated with the explicit transaction. This value must be mapped from the most recent pub.art.transaction:startTransaction that has not previously been committed or rolled back.

Output Parameters

None.

pub.art.transaction:setTransactionTimeout

This service enables you to manually set a transaction timeout interval for implicit and explicit transactions. When you use this service, you are temporarily overriding the Integration Server's transaction timeout interval. To change the server's default transaction timeout, see "Changing the Integration Server's Transaction Timeout Interval" on page 72.

You must call this service within a flow before the start of any implicit or explicit transactions. Implicit transactions start when you call an adapter service in a flow. Explicit transactions start when you call the pub.art.transaction:startTransaction service.

If the execution of a transaction takes longer than the transaction timeout interval, all current executions associated with the flow are cancelled and rolled back if necessary.

This service overrides only the transaction timeout interval for the flow service in which you call it.

Input Parameters

timeoutSeconds	**Integer.** The number of seconds that the implicit or explicit transaction stays open before the transaction manager aborts it.

Output Parameters

None.

pub.art.transaction:startTransaction

This service starts an explicit transaction. It must be used in conjunction with either a pub.art.transaction:commitTransaction service or pub.art.transaction:rollbackTransaction service. If it does not have a corresponding pub.art.transaction:commitTransaction service or.pub.art.transaction:rollbackTransaction service, your flow service will receive a runtime error.

For more information about implicit and explicit transactions, see "Transaction Management Overview" on page 68.

Input Parameters

startTransactionInput	**Document.** A document that contains the variable *transactionName*, described below.
transactionName	**String.** Specifies the name of the transaction to be started. This parameter is optional. If you leave this parameter blank, the Integration Server will generate a name for you. In most implementations, it is not necessary to provide your own *transactionName* as input.

Output Parameters

startTransactionOutput	**Document.** A document that contains the variable *transactionName*, described below.
transactionName	**String.** The name of the transaction the service just started.

Changing the Integration Server's Transaction Timeout Interval

The Integration Server's default transaction timeout is no timeout (NO_TIMEOUT). To change the server's transaction timeout interval, use a text editor to modify the server.cnf file and add the parameter below. Note that this parameter does not exist by default in the server.cnf file; you must add it to the file as described below.

Be sure to shut down the Integration Server before you edit this file. After you make changes, restart the server.

Add the following parameter to the server.cnf file:

```
watt.art.tmgr.timeout=TransactionTimeout
```

where *TransactionTimeout* is the number of seconds before transaction timeout.

This transaction timeout parameter does not halt the execution of a flow; it is the maximum number of seconds that a transaction can remain open and still be considered valid. For example, if your current transaction has a timeout value of 60 seconds and your flow takes 120 seconds to complete, the transaction manager will rollback all registered operations regardless of the execution status.

For more information about adding parameters to the server.cnf file, see the *Administering My webMethods Server* for your release.

Transaction Error Situations

When the Integration Server encounters a situation that could compromise transactional integrity, it throws an error. Such situations include the following:

- A transaction includes two or more different resources that support only local transactions.

  If a transaction accesses multiple resources, and more than one of the resources supports only local transactions, the integrity of the transaction cannot be guaranteed. For example, if the first resource commits successfully, and the second resource fails to commit, the first resource interaction cannot be rolled back; it has been committed already. To help prevent this problem, the Integration Server detects this case when connecting to more than one resource that does not support two-phase commits. It throws a run-time exception and the service execution fails.

  Note:
  Because this situation may be acceptable in some applications, the adapter user can include an input in the startTransaction service to cause the Integration Server to allow this situation.

- A resource is used in both a parent transaction and a nested transaction.

  This situation is ambiguous, and most likely means that a nested transaction was not closed properly.

- A parent transaction is closed before its nested transaction.

After a service request has invoked all its services, but before returning results to the caller, the service may commit its work. This commit could fail if the resource is unavailable or rejects the commit. This will cause the entire server request to fail and to roll back the transaction.