

TIBCO iWay® Service Manager

Software Development Kit User's Guide

Version 8.0 and Higher
March 2021
DN3502285.0321



Contents

1. Installing the iWay Software Development Kit	5
iWay Software Development Kit Overview	5
System Requirements	5
Installing the iWay SDK	5
Folder Structure	14
2. Getting Started With the iWay Software Development Kit	17
Understanding Apache Ant Tasks	17
iwaddtemplate.....	18
iwbuild.....	18
iwdelete.....	19
iwdeploy.....	20
iwdeploylocal.....	21
iwscript.....	22
iwstart.....	23
iwstop.....	24
iwupload.....	25
Using Sample Integration Tasks	26
Creating Web Archives (WAR) Files	28
Using the iWay SDK	29
Legal and Third-Party Notices	31

Installing the iWay Software Development Kit

This section provides an introduction for the iWay Software Development Kit (SDK) and describes the system requirements that are needed for installation.

In this chapter:

- [iWay Software Development Kit Overview](#)
 - [System Requirements](#)
 - [Installing the iWay SDK](#)
 - [Folder Structure](#)
-

iWay Software Development Kit Overview

The iWay Software Development Kit (SDK) provides the tools and technologies that are required for the development of iWay applications and their web archives at an enterprise level.

System Requirements

The iWay SDK requires the following software components to be installed on your system and configured appropriately:

- Java Version 1.8 or higher
- Apache Ant Version 1.7.1 or higher
- Ant-Contrib Tasks

For more information, see the following website:

<http://ant-contrib.sourceforge.net/>

- iWay Service Manager (iSM) Version 8.0

Note: For deployment and application start and stop tasks, an iSM server-side enhancement is required to ensure proper operation.

Installing the iWay SDK

This section describes how to install the iWay SDK on your system.

Follow the procedure for your platform:

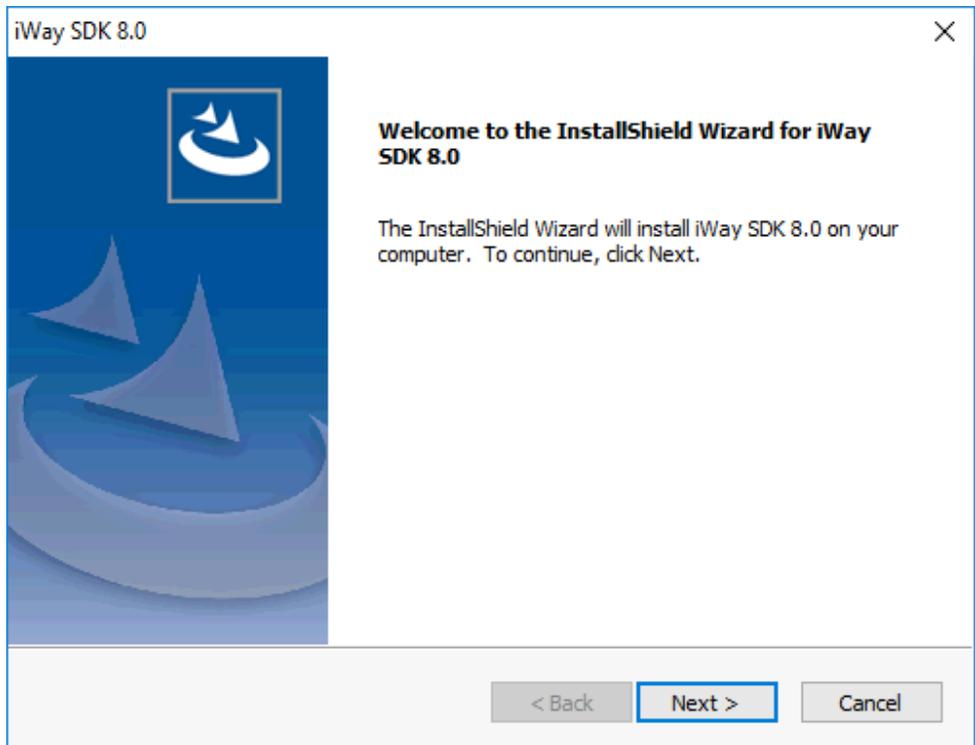
- For Windows, see [How to Install the iWay SDK on Windows](#) on page 6.
- For UNIX, OS/400, and z/OS, see [How to Install the iWay SDK on UNIX, OS/400, and z/OS](#) on page 11.

Procedure: How to Install the iWay SDK on Windows

You must be an administrator for the local machine to run the installation. To install iWay SDK on Windows:

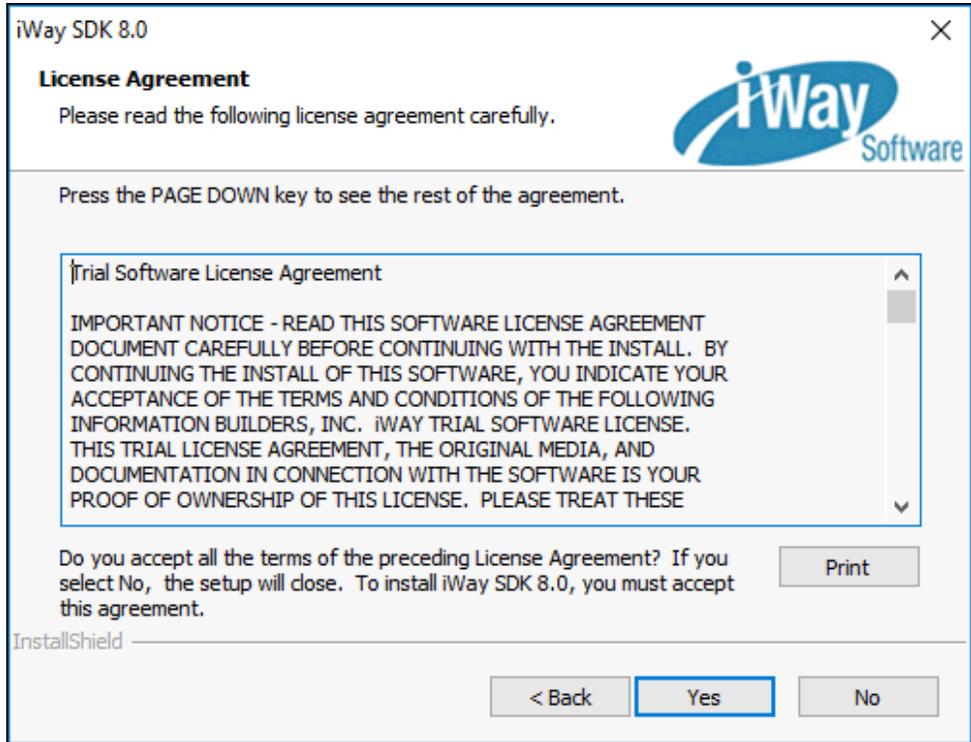
1. Download and execute the iwSDK80.exe file.

The iWay SDK 8.0 Welcome window opens, as shown in the following image.



2. Click Next.

The License Agreement window for the iWay SDK opens, as shown in the following image.



3. Review the license agreement, and click Yes if you agree to the terms and want to continue with the iWay SDK installation.

The Customer Information window opens, as shown in the following image.

iWay SDK 8.0

Customer Information
Please enter your information.

iWay Software

User Name:
Information Builders

Company Name:
Information Builders

Site Code: Enter 9999 for installing as a trial version.
9999

Install this application for:

Anyone who uses this computer (all users)

Only for me (Information Builders)

InstallShield

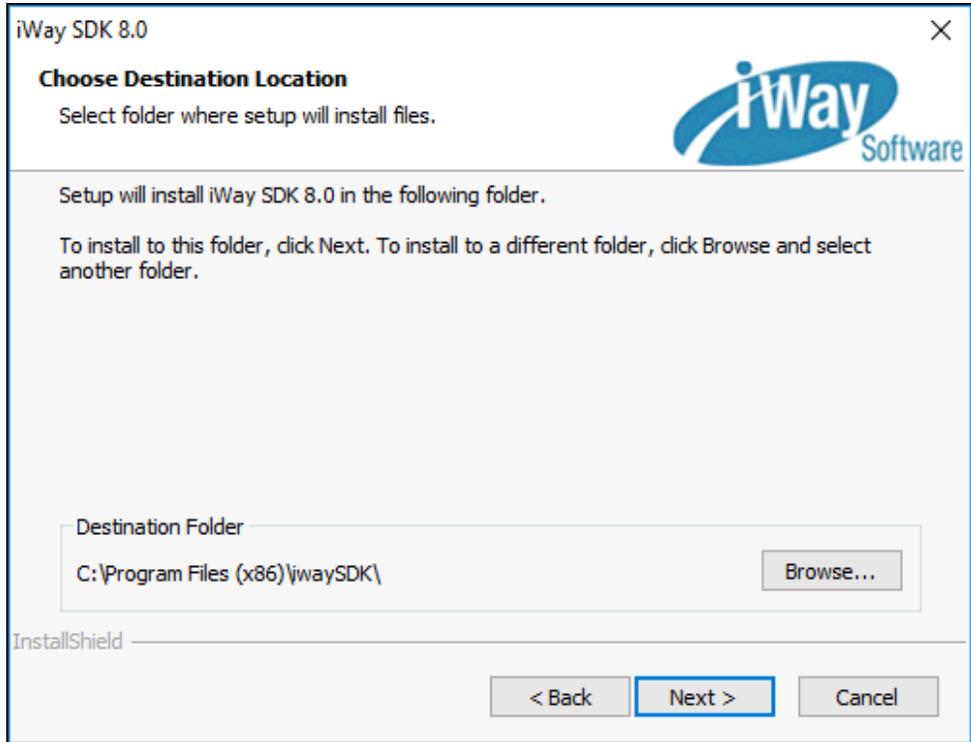
< Back Next > Cancel

4. Provide your *User Name*, *Company Name*, and *Site Code*.

Important: The site code is a unique company identifier associated with a specific machine. Ensure to enter a valid and accurate site code in this step because this entry is used when generating your permanent license during the registration process. If you need assistance with the site code, contact your iWay Software sales representative.

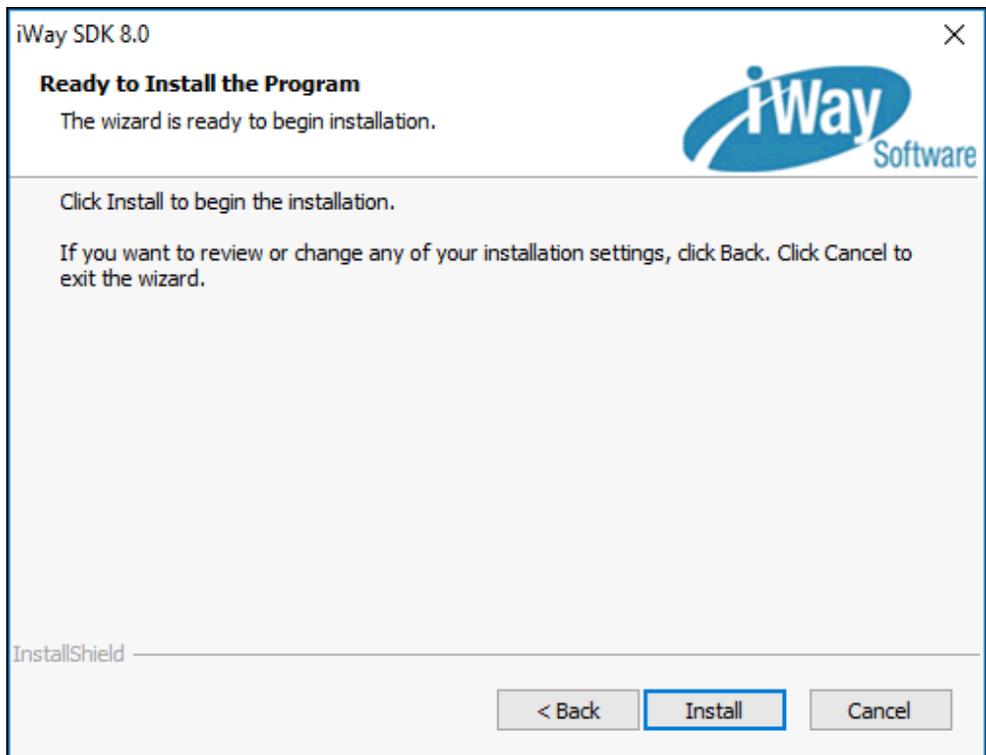
5. Click *Next*.

The Choose Destination Location window opens, as shown in the following image.



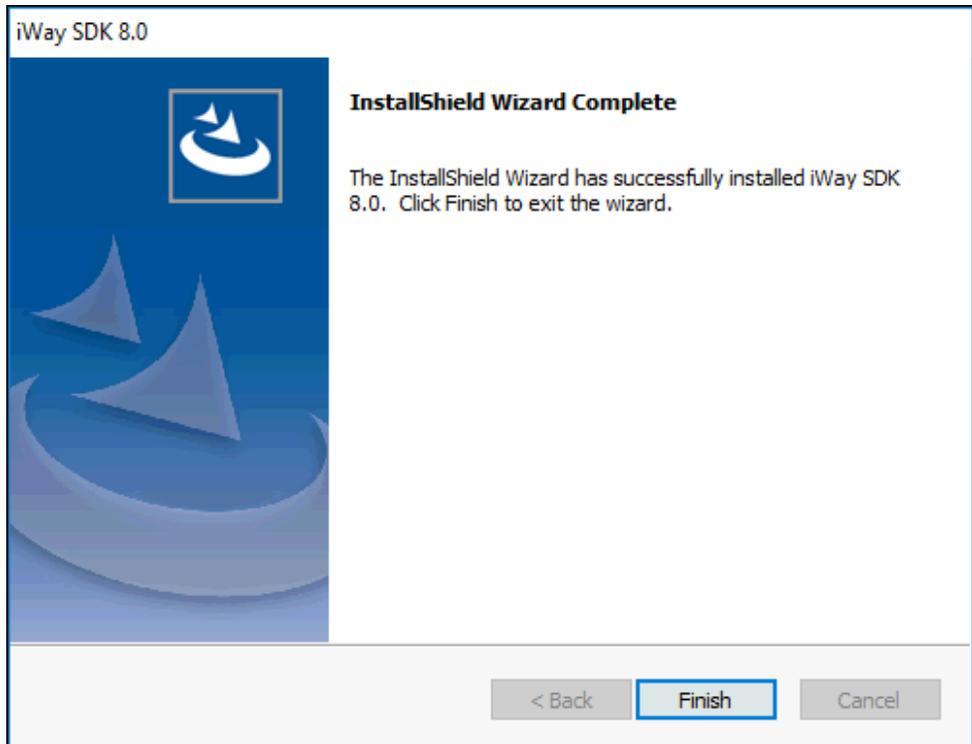
6. Accept the default or click *Browse* to specify a new location. Then, click *Next*.

The Ready to Install the Program window opens, as shown in the following image.



7. Click *Install*.

After the iWay SDK installation has finished, the following window is displayed.



8. Click *Finish*.

Procedure: How to Install the iWay SDK on UNIX, OS/400, and z/OS

On UNIX/Linux, installing as root is not recommended. Creating a dedicated iWay user and group with appropriate rights is preferable.

On OS/400, your user ID must have *ALLOBJ, *JOBCTL, and *SAVSYS authority.

On z/OS, the iwSDK80.jar file must be placed in the USS file system.

The new unified iWay installer can enable silent, unattended installation. For more information, contact iWay Customer Support.

1. Use FTP in binary mode to transfer the iwSDK80.jar file to your UNIX or OS/400 machine. For OS/400, place the iwSDK80.jar file in a directory under QSH.
2. Navigate to the directory containing the iwSDK80.jar file. On OS/400, you must be running under QSH.

3. Ensure the installation file is executable, for example:

```
chmod 755 iwSDK80.jar
```

4. Start the installation by executing:

```
java -jar iwSDK80.jar
```

The iWay SDK installation initializes, which may take some time. When initialization is complete, a Welcome prompt appears:

```
Welcome to the iWay SDK Setup Wizard. This setup program installs iWay
SDK 8.0.0.101
```

```
Setup is using Windows 10 10.0 amd64 Settings
File encoding is Cp1252, XML encoding is UTF-8
```

```
Copyright (C) 2009-2017, iWay Software/information Builders, Inc.
All Rights Reserved.
```

```
Press 1 for Next, 2 to Cancel [1]
```

Note: If the installation does not launch, ensure that `/JAVA_HOME/bin` is in your `$PATH` variable.

5. Press Enter to continue.

A license agreement appears.

6. Review the agreement and press Enter until you see the following prompt:

```
Please choose from the following options:
```

```
[ ] 1 - I accept the terms of the license agreement.
[X] 2 - I do not accept the terms of the license agreement.
```

```
To select an item enter its number, or 0 when you are finished: [0]
```

7. If you accept the terms, type 1 and press Enter.

The prompt repeats showing the new value.

```
[X] 1 - I accept the terms of the license agreement.
[ ] 2 - I do not accept the terms of the license agreement.
```

```
To select an item enter its number, or 0 when you are finished: [0]
```

8. Type 0, then press Enter.

The installation directory prompt appears:

```
Destination Location
```

Setup will install iWay SDK in the following location. Setup allows users to enter a different location.

```
Directory: [/iwaySDK/8.0.0]
```

Note: On Linux systems, you may need to change the default directory that appears. The default directory normally should be named iWaySDK, but some Linux environments do not follow this default.

- Specify where to install iWay on your system and then press Enter. Ensure this is a directory to which you have write access.

The navigation prompt appears.

```
Press 1 for Next, 2 for Previous, 3 to Cancel or 4 to Redisplay [1]
```

- Press Enter to continue with the installation.

A notice and summary are displayed to inform you that you have provided enough information to start copying files.

```
Start Copying Files
```

Setup has enough information to start copying the program files. If you want to review or change any settings, now is the time to do so.

```
iWay SDK will be installed in the following location:
/iwaySDK/8.0
```

```
Press 1 for Next, 2 for Previous, 3 to Cancel or 4 to Redisplay [1]
```

- Press Enter to start the installation.

The Setup Status information is displayed.

```
Setup Status
```

Setup is configuring your new software installation.

```

|-----|-----|-----|-----|
0%      25%    50%    75%    100%
||||||||||||||||||||||||||||||||
```

Once installation has finished, a message appears indicating that the setup is complete.

```
Setup Completes
```

Setup has finished installing iWay SDK on your system.

```
Press 1 to Finish Setup [1]
```

12. Press Enter to finish the installation.

Folder Structure

This section describes the folder structure that is created after you install the iWay SDK on your system.

\ant

Contains the tools.xml file, which is an Ant script file that the iWay SDK includes for enhanced ANT script patterns.

\bin

Contains the install.xml file, which contains information obtained during the installation and includes versioning information for the iWay SDK.

\build

This folder utilizes ANT tasks found in the iwscript.jar file, which is located in the \lib directory. The following files are included in the \build folder:

build.cmd (Windows), build.sh (UNIX). Performs build type tasks.

build.xml. Ant (interface) for build.cmd or build.sh.

iwbuild.xml. Ant support file for build.xml utilizing the iwscript Ant interface.

The \build folder contains a \configurations subfolder, which contains a set of build/deploy configurations utilized by build.xxx found within the build directory (build\configurations\).

<sample_config>\default\ (target configuration assets)

Contains the following subfolders and files:

\war. This folder includes custom WEB-INF and META-INF directories to be merged during the BUILDWAR task.

\scripts. Contains the user.xml file. This is an Ant-based script file used to customize the pre-execution and post-execution of supported tasks.

\dist. Serves as the build destination for iIA and WAR deployments.

- ❑ `default.properties (target)`. The Ant property file used to manage build and deployment options as name-value pairs. The following structure for maintained configuration is expected:

- ❑ `newconfiguration`
 - ❑ `newconfiguration` directory
 - ❑ `default.properties`
 - ❑ `customtarget.properties`
 - ❑ `customtarget1.properties`

The `\build` folder also contains a `\projects` subfolder, which is the location of iIT Eclipse-based projects. The `\projects` subfolder contains a sample application (iIT project) under `\app` sample IIT project which can be used to test the iWay SDK.

- ❑ `\config`

The iWay Service Manager configuration directory.

Note: Do not alter the contents within this directory.

- ❑ `\etc`

Contains the following subfolders:

- ❑ `\doc`

Contains `iwscript` Java documentation in the `etc\doc\iwscript\java` folder. Contains `iwscript` Ant documentation in the `etc\doc\iwscript\ant` folder. Contains the iWay SDK build documentation in the `etc\doc\build` folder.

- ❑ `\licenses`

Contains license files for the iWay SDK, including those required by third-party open source distributions.

- ❑ `\manager`

The directory of the deployment, which contains the `deployment\jia` subfolder. The `iwscript-ant-tasks.xml` file is located here, which contains the code for the Ant `iwscript` interface.

- ❑ `\packages`

This is a required empty directory.

- ❑ `\setup`

Contains the ismbase.war file, which is a sample .war file packaged with iSM server resources.

□ \lib

Contains the iwscript.jar, which is the Ant interface used to build and deploy iWay applications. The remaining .jar files in this folder are supporting files for the iwscript.jar file.

This section describes how to configure and use the iWay Software Development Kit (SDK).

In this chapter:

- [Understanding Apache Ant Tasks](#)
 - [Using Sample Integration Tasks](#)
 - [Creating Web Archives \(WAR\) Files](#)
 - [Using the iWay SDK](#)
-

Understanding Apache Ant Tasks

The iWay SDK is made up of two major components. The first, an Apache Ant extension that exposes several tasks for managing the building and deploying process for an iWay Integration Application (iIA).

The following is a list of Ant tasks that are currently supported by the iWay SDK:

- iwaddtemplate.** Uploads an application template file (.ita) to iWay Service Manager (iSM).
- iwbuild.** Builds an iWay Integration Application (.iia file) from an Eclipse-based iIT project. This task can only run in an application directory, ending with *.iab.
- iwdelete.** Deletes an iIA template (.ita), application (.iia), or deployment from iSM.
- iwdeploy.** Deploys an iIA with a specified template (.ita) to iSM.
- iwdeploylocal.** Deploys an iIA with a specified template (.ita) to a local directory.
- iwscript.** Executes a remote Ant script through iSM.
- iwstart.** Starts a deployed iIA or application channel(s). If no channel nodes are found, then the application is started. Otherwise, specified channels are started.
- iwstop.** Stops a deployed iIA or application channel(s). If no channel nodes are found, then the application is stopped. Otherwise, specified channels are stopped.
- iwupload.** Uploads an iIA archive file (.iia) to iSM.

These Ant tasks provide a rich feature set that can assist build masters to integrate iWay into their new or existing software manufacturing systems.

iwaddtemplate

Uploads an application template file (.ita) to iWay Service Manager (iSM). The template will be renamed into templateName.

Parameters:

The following table lists and describes the parameters for the iwaddtemplate Ant task.

Attribute	Description	Required
fileName	File path to the template file (.ita).	yes
templateName	Name of the template on iSM.	yes
user	A valid user name that is used to connect to iSM.	yes
password	A valid password that is used to connect to iSM.	yes
serverURL	The URL used to access iSM.	yes

Example:

The following build.xml snippet is an example of how to invoke the iwaddtemplate Ant task:

```
<property name="new.template.name" value="uploaded" />
<property name="template.file.name" value="../dev.ita" />
<property name="server.url" value="http://localhost:9000" />
<property name="server.user" value="iway"/>
<property name="server.password"
value="ENCR(3237324531043128310632252993121)" />
<iwaddtemplate
  templateName="${new.template.name}"
  fileName="${template.file.name}"
  serverURL="${server.url}"
  userName="${server.user}"
  password="${server.password}"
/>
```

iwbuild

Builds an iWay Integration Application (.iia file) from an Eclipse-based iIT project. This task can only run in an application directory, ending with *.iab.

Parameters:

The following table lists and describes the parameters for the `iwbuild` Ant task.

Attribute	Description	Required
<code>dir</code>	The file path to the application directory, which must end with <code>*.iab</code> .	no
<code>sdkPath</code>	A list of additional workspaces separated by a semicolon (;). Workspaces are directories that contain iIT projects. Set this attribute if your application contains iIT project components in other workspaces.	no
<code>clean</code>	Set this attribute to <code>true</code> if you want to recompile the artifacts. This attribute is set to <code>false</code> by default.	no

Example:

The following `build.xml` snippet is an example of how to invoke the `iwbuild` Ant task:

```
<iwbuild dir="${appdir}" sdkPath="${otherworkspace}" />
```

iwdelete

Deletes an iIA template (.ita), application (.iia), or deployment from iSM.

Parameters:

The following table lists and describes the parameters for the `iwdelete` Ant task.

Attribute	Description	Required
<code>name</code>	Name of the iIA template, application, or deployment.	yes
<code>type</code>	The resource type (<i>app</i> , <i>deployment</i> , or <i>template</i>). This attribute is set to <i>deployment</i> by default.	no
<code>user</code>	A valid user name that is used to connect to iSM.	yes

Attribute	Description	Required
password	A valid password that is used to connect to iSM.	yes
serverURL	The URL used to access iSM.	yes

Example:

The following build.xml snippet is an example of how to invoke the iwdelete Ant task:

```
<property name="deployment.name" value="from_ant" />
  <property name="server.url" value="http://localhost:9000" />
  <property name="server.user" value="iway"/>
  <property name="server.password"
value="ENCR( 3237324531043128310632252993121)"/>
<iwdelete
  name="${deployment.name}"
  serverURL="${server.url}"
  userName="${server.user}"
  password="${server.password}"
/>
```

iwdeploy

Deploys an iIA with a specified template (.ita) to iSM.

Parameters:

The following table lists and describes the parameters for the iwdeploy Ant task.

Attribute	Description	Required
app	Name of the application in iSM that you want to deploy.	yes
templateName	Name of the template in iSM that you want to deploy.	yes
deploymentName	The application deployment name, which defaults to app_templateName.	yes
port	Console port for the application. If a port value is not set, the next available port will be assigned.	no
user	A valid user name that is used to connect to iSM.	yes

Attribute	Description	Required
password	A valid password that is used to connect to iSM.	yes
serverURL	The URL used to access iSM.	yes

Example:

The following build.xml snippet is an example of how to invoke the iwdeploy Ant task:

```
<property name="app" value="app" />
  <property name="template.name" value="raw" />
  <property name="server.url" value="http://localhost:9000" />
  <property name="server.user" value="iway" />
  <property name="server.password"
value="ENCR(3237324531043128310632252993121)" />
<iwdeploy
  app="${app}"
  templateName="${template.name}"
  serverURL="${server.url}"
  userName="${server.user}"
  password="${server.password}"
/>
```

iwdeploylocal

Deploys an iIA with a specified template (.ita) to a local directory without running iSM.

Parameters:

The following table lists and describes the parameters for the iwdeploylocal Ant task.

Attribute	Description	Required
appDir	Application directory, ending with *.iab.	yes
sdkHome	Directory where iWay SDK is installed.	yes
templateFile	Location of the template (.ita) file.	yes
override	Set this attribute to <i>true</i> if you want to override the existing local deployment (if it exists). This attribute is set to <i>false</i> by default.	no

Example:

The following build.xml snippet is an example of how to invoke the iwdeploylocal Ant task:

```
<property name="appDir" value="C:\iway\src\8.0\components\iwscrip\ttestdata
\projects\app\Applications\mover.iab" />
  <property name="template.file" value="C:\iway\src\8.0\components
\iwscrip\ttestdata\projects\app\dev.ita" />
  <iwdeploylocal
    sdkHome="${basedir}"
    appDir="${appDir}"
    templateFile="${template.file}"
    override="true"
  />
```

iwscrip

Executes a remote Ant script through iSM.

Parameters:

The following table lists and describes the parameters for the iwscrip Ant task.

Attribute	Description	Required
antTarget	The Ant target to invoke inside the Ant script.	yes
app	The name of the application deployment.	yes
script	The script name, which is usually build.xml.	yes
user	A valid user name that is used to connect to iSM.	yes
password	A valid password that is used to connect to iSM.	yes
serverURL	The URL used to access iSM.	yes

Example:

The following build.xml snippet is an example of how to invoke the iwscrip Ant task:

```

<property name="remote.target" value="props" />
  <property name="deployment.name" value="from_ant" />
  <property name="server.url" value="http://localhost:9000" />
  <property name="server.user" value="iway" />
  <property name="server.password"
value="ENCR(3237324531043128310632252993121)" />

  <iwscript
    serverurl="${server.url}"
    anttarget="${remote.target}"
    app="${deployment.name}"
    script="build.xml"
    username="${server.user}"
    password="${server.password}"
  />

```

iwstart

Starts a deployed iIA or application channel(s). If no channel nodes are found, then the application is started. Otherwise, specified channels are started.

Parameters:

The following table lists and describes the parameters for the iwstart Ant task.

Attribute	Description	Required
deploymentName	The name of the application deployment.	yes
user	A valid user name that is used to connect to iSM.	yes
password	A valid password that is used to connect to iSM.	yes
serverURL	The URL used to access iSM.	yes
timeout	Determines the amount of time (in seconds) to wait for an application deployment or channel to start. The default value is 10 seconds.	no
failonerror	If set to <i>true</i> , the execution of the script will terminate if an application deployment or a channel fails to start before timeout occurs.	no

Examples:

The following build.xml snippet is an example of how to start an application deployment:

```

<property name="deployment.name" value="from_ant" />
<property name="server.url" value="http://localhost:9000" />
<property name="server.user" value="iway"/>
<property name="server.password"
value="ENCR(3237324531043128310632252993121)" />
<iwstart
  deploymentName="${deployment.name}"
  serverURL="${server.url}"
  userName="${server.user}"
  password="${server.password}"
/>

```

The following build.xml snippet is an example of how to start three channels (file1, file2, and file3):

```

<iwstart
  deploymentName="${deployment.name}"
  serverURL="${server.url}"
  userName="${server.user}"
  password="${server.password}">
  <Channel name="file1"/>
  <Channel name="file2"/>
  <Channel name="file3"/>
</iwstart>

```

iwstop

Stops a deployed iIA or application channel(s). If no channel nodes are found, then the application is stopped. Otherwise, specified channels are stopped.

Parameters:

The following table lists and describes the parameters for the iwstop Ant task.

Attribute	Description	Required
deploymentName	The name of the application deployment.	yes
user	A valid user name that is used to connect to iSM.	yes
password	A valid password that is used to connect to iSM.	yes
serverURL	The URL used to access iSM.	yes
timeout	Determines the amount of time (in seconds) to wait for an application deployment or channel to stop. The default value is 10 seconds.	no

Attribute	Description	Required
failonerror	If set to <i>true</i> , the execution of the script will terminate if an application deployment or a channel fails to stop before timeout occurs.	no

Examples:

The following build.xml snippet is an example of how to stop an application deployment:

```
<property name="deployment.name" value="from_ant" />
<property name="server.url" value="http://localhost:9000" />
<property name="server.user" value="iway"/>
<property name="server.password"
value="ENCR( 3237324531043128310632252993121) " />
<iwstop
  deploymentName="${deployment.name}"
  serverURL="${server.url}"
  userName="${server.user}"
  password="${server.password}"
/>
```

The following build.xml snippet is an example of how to stop three channels (file3, file2, and file1):

```
<iwstop
  deploymentName="${deployment.name}"
  serverURL="${server.url}"
  userName="${server.user}"
  password="${server.password}">
  <Channel name="file3"/>
  <Channel name="file2"/>
  <Channel name="file1"/>
</iwstop>
```

iwupload

Uploads an application (.iia) file to iSM.

Parameters:

The following table lists and describes the parameters for the iwupload Ant task.

Attribute	Description	Required
applicationFile	The file path to the application (.iia) file.	no
user	A valid user name that is used to connect to iSM.	yes

Attribute	Description	Required
password	A valid password that is used to connect to iSM.	yes
serverURL	The URL used to access iSM.	yes

Example:

The following build.xml snippet is an example of how to invoke the iwupload Ant task:

```
<property name="iia.name" value="C:\app.iia" />
  <property name="template.name" value="raw" />
  <property name="server.url" value="http://localhost:9000" />
  <property name="server.user" value="iway" />
  <property name="server.password"
value="ENCR(3237324531043128310632252993121)" />
<iwupload
  applicationFile="${iia.name}"
  serverURL="${server.url}"
  userName="${server.user}"
  password="${server.password}"
/>
```

Using Sample Integration Tasks

iWay SDK provides sample integration tasks that can be used for building and deploying iIAs. The *iWaySDKHome*\build directory contains sample build scripts that support the following integration tasks:

- ABOUT.** Displays Help topics about a specific task.
- BUILDAPP.** Builds an iWay Integration Application (iIA) from an Eclipse-based iIT project.
- BUILDWAR.** Builds a Web Archive (WAR) based on a iIA and template file.
- DEPLOYAPP.** Deploys an iIA to a local or remote iSM server.
- UNDEPLOYAPP.** Stops and undeploys an iIA.
- UPDATEAPP.** Updates an iIA on a local or remote server.
- STARTAPP.** Starts a local or remote application.
- STOPAPP.** Stops a local or remote application.

The sample integration tasks must be executed by the build utility, which is located in the *iWaySDKHome*\build directory. You can review the list of these tasks by typing the following command in the command window:

```
build ABOUT
```

For more information on specific build task, type the following command:

```
build ABOUT <TASKNAME>
```

You can invoke the tasks by typing the following command in the command window:

```
build.cmd TASKNAME CONFIGURATION <TARGETNAME>
```

where:

TASKNAME

Is the name of the build integration task.

CONFIGURATION

Is the build configuration located under *iWaySDKHome*\build\configurations.

TARGET

Is the optional name of the target properties file, which defaults to default.properties.

ABOUT

Describes help topics about a specified task.

A sample configuration called *ipay* is packaged with the iWay SDK to demonstrate each of the sample integration tasks.

To begin, type the following command in the command window:

```
build BUILDAPP ipay
```

This will execute the BUILDAPP task and build the iWay Integration Application (iIA) defined in the default target of the *ipay* configuration. Configurations are located in the \build\configurations folder. For example, browse to the following file:

```
\build\configurations\ipay\default.properties
```

The default.properties file contains a rich set of configuration properties that drive the build and deployment process. Documentation for these properties can be found by studying the comments found in this file or by typing the following command in the command window:

```
build ABOUT BUILDAPP
```

The following information is displayed:

BUILDAPP. Builds iWay Integration Applications (iIAs) from iIT Eclipse-based projects.

- ❑ Configuration properties:
 - ❑ **application.name.** The iIA to build.
 - ❑ **iitproject.name.** The iIT project name where the iIA exists.

This can be repeated for each of the TASKS of the build system. Moving forward, notice that after executing BUILDAPP using the iway configuration and default target, that the iWay SDK has created the following:

```
\build\configurations\iway\default\dist\mover.iaa
```

The build assumes that the sources (iIT projects) defining iIAs are located in the following directory:

```
\build\projects\
```

Regardless whether the iIT project references a project from a different Eclipse workspace, all dependent iIT projects must be available in this directory.

To deploy mover.iaa to an iSM server, enter the following:

```
build DEPLOYAPP iway
```

Endpoints, including authentication information (user ID and password), can once again be found in the property file for the target (for example, default.properties).

To start or stop the iIA, use the STARTAPP or STOPAPP integration tasks.

Creating Web Archives (WAR) Files

The iWay SDK is packaged with the sample configuration called iway, which contains two targets:

- ❑ **default.** Configured for building WAR files for application servers using the web.xml 1.4 specification.
- ❑ **WASCE.** Configured for building WAR files for IBM WebSphere Application Server Community Edition (WASCE).

If you look at each of these targets, you will notice that there exists one or more deployment descriptors in the following directory:

```
\build\configuration\target_name\war\WEB-INF
```

For most application servers, a single descriptor file (web.xml) is required. For WASCE however, an additional file called geronimo web.xml is required. Consult the documentation for the application server for its descriptor format and requirements.

To demonstrate WAR creation, enter:

```
build BUILDWAR iway
```

or

```
build BUILDWAR iway WASCE
```

A mover_dev.war file will appear in the `\build\configuration\target_name\dist` directory. This file can now be deployed into an application server. For more information on deploying WAR files, see the documentation for the application server.

Once deployed, invoking the application will display the ISM console license page indicating that the application is not authorized to run within an application server container. The iWay SDK is not packaged with a license file with this functionality enabled. There are two workarounds. The first is to request a license file for the SDK with this feature enabled from an iWay Software Customer Support representative.

With a new license file in hand, copy the file into the root directory of the SDK. Then, in the target configuration file (for example, `\build\configuration\target_name.properties`), uncomment the following property:

```
update.license
```

This will now insert the new license file into the WAR file. Redeploy the WAR file and the application will be authorized for servlet functionality.

Another method of averting the license issue is to override the default WAR source file. By default, the iWay SDK uses the following WAR file as its base for iSM server components:

```
\etc\setup\ismbase.war
```

WAR files generated by iSM servers authorized for servlet deployment will contain the proper licensing, which can be used accordingly. To override the iWay SDK default WAR file, refer back to the configuration of the target and set the following property accordingly:

```
warsource.war=c:\\customwar.war
```

Providing customwar.war is authorized to run in an application server. Any WAR file generated through BUILDWAR using this target will also do the same.

Using the iWay SDK

This section demonstrates how to use the iWay SDK. A best practice is to copy the contents of the `\build\configuration\iway` configuration to a new configuration (for example, mynewconfig).

At this point, you should have a `\build\configuration\mynewconfig` directory with a `default.properties` file. Open the `default.properties` file in a text editor. The top of the file should look something like the following:

```
#####  
# IIT Project Properties  
#####  
# Name of IIT project found within the projects directory which contains  
# the application component iitproject.name=app  
# Name of IIT application component found with the selected IIT project  
application.name=mover
```

Perform the following steps:

1. Change the property called `iitproject.name` to equal the name of your iIT project.
2. Modify the property called `application.name` to the name of the application in the iIT project you want to build
3. Copy the iIT project and its dependant projects (if any) to the following directory:

```
\build\projects
```

4. Enter the following in the build directory:

```
build BUILDAPP mynewconfig
```

If your application does not have any errors or missing dependencies, there will be an `.iia` file created in the following directory:

```
\build\configuration\mynewconfig\dist
```

The iWay SDK sample build process also offers prologues and epilogues for each of its build tasks. To demonstrate how to hook user written ANT tasks into the build process, refer to the following file:

```
\build\configurations\iway\default\scripts\user.xml
```

This file contains tasks for each of the supported build tasks. For example:

```
<target name="buildapp_prologue" >  
<echo>===== user:buildapp_prologue</echo>  
</target>  
<target name="buildapp_epilogue" >  
<echo>===== user:buildapp_epilogue</echo>  
</target>
```

With this file in place, these targets will execute before and after each selected task.

Legal and Third-Party Notices

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of TIBCO Software Inc.

TIBCO, the TIBCO logo, the TIBCO O logo, FOCUS, iWay, Omni-Gen, Omni-HealthData, and WebFOCUS are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle Corporation and/or its affiliates.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the readme file for the availability of this software version on a specific operating system platform.

THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

This and other products of TIBCO Software Inc. may be covered by registered patents. Please refer to TIBCO's Virtual Patent Marking document (<https://www.tibco.com/patents>) for details.

Copyright © 2021. TIBCO Software Inc. All Rights Reserved.