Multichannel Bank Transformation Toolkit
Version 8.0

Migrating from WebSphere Multichannel Bank Transformation Toolkit version 5.2 to WebSphere Multichannel Bank Transformation Toolkit version 8.0

IBM

Note!

Before using this information and the product it supports, be sure to read the general information under "Notices" on page 23.

This edition applies to Version 8, Release 0, Modification 0, of *IBM WebSphere Multichannel Bank Transformation Toolkit* (5724-H82) and to all subsequent releases and modifications until otherwise indicated in new editions.

IBM welcomes your comments. You can send to the following address:

IBM China Software Development Lab
Bank Transformation Toolkit Product
Diamond Building, ZhongGuanCun Software Park, Dongbeiwang West Road No.8,
ShangDi, Haidian District, Beijing 100193 P. R. China

Include the title and order number of this book, and the page number or topic related to your comment.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Migrating from WebSphere Multichannel Bank Transformation Toolkit version 5.2 to version 8.0

This section provides information that will enable users of IBM® WebSphere® Multichannel Bank Transformation Toolkit version 5.2 move up to the full power and versatility of IBM WebSphere Application Server. It provides the migration path to allow an enterprise to reuse much of its existing toolkit application code base and still take advantage of the WebSphere J2EE environment.

IBM recognizes that an enterprise cannot always upgrade its application software all at once. Because the enterprise can decide how much or how little toolkit functionality to retain, WebSphere Multichannel Bank Transformation Toolkit version 8.0 enables the enterprise to upgrade to a full J2EE environment incrementally. With WebSphere Multichannel Bank Transformation Toolkit version 8.0, the enterprise can bypass the toolkit functionality and access J2EE components directly upon requests.

WebSphere Multichannel Bank Transformation Toolkit version 8.0 provides a migration tool that helps you to migrate your applications. The migration tool can migrate WebSphere Multichannel Bank Transformation Toolkit version 5.2 definitions to WebSphere Multichannel Bank Transformation Toolkit version 8.0 definitions and migrate WebSphere Multichannel Bank Transformation Toolkit version 5.2 class packages and APIs to the corresponding WebSphere Multichannel Bank Transformation Toolkit version 8.0 class packages and APIs.

This section describes the concepts and tasks that are related to migrating your applications to WebSphere Multichannel Bank Transformation Toolkit version 8.0 with the migration tool.

Concepts

Before you perform the migration tasks, it is recommended that you understand the concepts that are involved in the migration process. For detailed information on the concepts that are related to migrating from IBM WebSphere Multichannel Bank Transformation Toolkit version 5.2 to IBM WebSphere Multichannel Bank Transformation Toolkit version 8.0, refer to the following sections.

Migration Overview

IBM WebSphere Multichannel Bank Transformation Toolkit version 8.0 provides the following functions to help you migrate your applications from WebSphere Multichannel Bank Transformation Toolkit version 5.2 to WebSphere Multichannel Bank Transformation Toolkit version 8.0:

Definition transformation

This feature transforms the definition files from WebSphere Multichannel Bank Transformation Toolkit version 5.2 to WebSphere Multichannel Bank Transformation Toolkit version 8.0. All the existing definition files are converted or migrated to the corresponding WebSphere Multichannel Bank Transformation Toolkit version 8.0. For example, the `dse.ini` definition file is converted to the `btt.xml` file.

Java and JSP code migration

The migration tool scans the code of your applications and automatically replaces the code of WebSphere Multichannel Bank Transformation Toolkit version 7.0, or earlier, with WebSphere Multichannel Bank Transformation Toolkit version 8.0 code by renaming packages and some classes and methods. This reduces the workload for the application program migration.

Default migration rules provided

WebSphere Multichannel Bank Transformation Toolkit version 8.0 provides default migration rules for migration from WebSphere Multichannel Bank Transformation Toolkit version 5.2 to WebSphere Multichannel Bank Transformation Toolkit version 8.0. You can use the default rules or extend the default rules to meet additional migration requirements.

Customer extension support

The migration tool is rule-based, which means that the migration tasks that are carried out are defined as rules. You can add new rules to meet your application migration requirements.

Exclusions

UI features here refer to the and Swing-based presentation logic. In WebSphere Multichannel Bank Transformation Toolkit version 8.0, Swing-based Java client is wholly compatible with WebSphere Multichannel Bank Transformation Toolkit version 5.2; therefore, no migration is required.

Constraints

To reduce the complexity of the migration process, the number of tasks that are performed by the migration tool are limited by the rules that are defined in the tool. Migration tasks that are not defined in the migration tool as rules are not performed.

Benefits gained through the migration

IBM WebSphere Multichannel Bank Transformation Toolkit version 8.0 provides support for distributed architecture. Adopting a distributed architecture provides the benefits to both system planners and developers.

Migration provides the following benefits:

- Scalability improves by an order of magnitude. A version 8.0 system can scale from a small or medium enterprise having a single server to large enterprises with server farms distributed across multiple regional data centers.
- Performance can be tuned by balancing hardware and network usage against the expected user base. IBM Rational® Application Developer and IBM WebSphere Integration Developer provide the tools and guides for performing this tuning.
- Industry standards such as JCA, EJB, JMS, and Web services replace many previous architectures and facilities in the toolkit. This adds flexibility to expanding enterprise applications using third-party components that are built with the same industry standard architectures.
- To enhance its flexibility, the toolkit maintains much of its previous external design philosophy. It further adds more configurability to this mix by allowing its deployment information to be configured. Scaling and performance tuning are all performed using configuration settings rather than Java code changes.

Migration tool architecture and support

IBM WebSphere Multichannel Bank Transformation Toolkit version 8.0 provides a migration tool to help you migrate the applications that you developed with pervious versions of the toolkit to the WebSphere Multichannel Bank Transformation Toolkit version 8.0 architecture.

A high scalability architecture has been applied to this rule-based migration tool. Under this architecture, you can easily extend the toolkit to meet your specific requirements. The following figure shows the architecture of the WebSphere Multichannel Bank Transformation Toolkit rule-based migration tool.

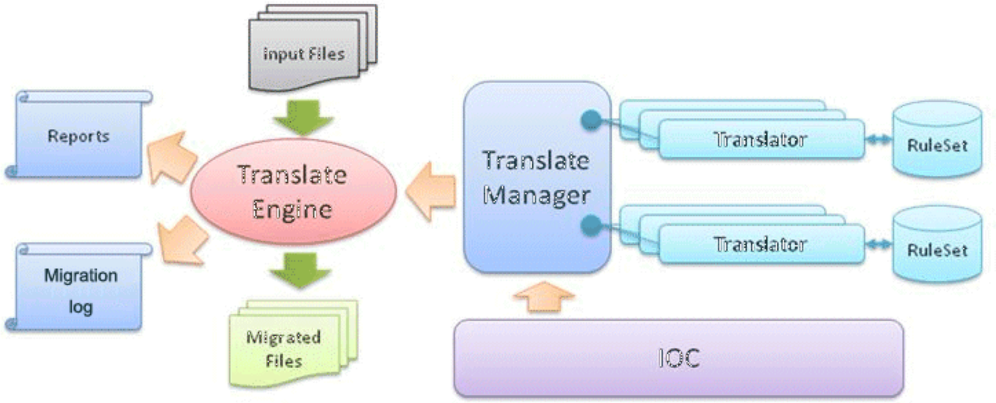

Figure 1. Migration tool architecture

The functional descriptions of the migration tool are provided in the following list:

- Migrate Java and JSP code:

  From WebSphere Multichannel Bank Transformation Toolkit version 5.2 to WebSphere Multichannel Bank Transformation Toolkit version 8.0, there are API changes to the same functions. It is time-consuming to replace the old API calls with the new ones. The code migration function of the migration tool enables Java and JSP code to be migrated automatically.
- Migrate the `dse.ini` file:

  The `dse.ini` file contains most of the configurations for the components of WebSphere Multichannel Bank Transformation Toolkit version 5.2. New components have been added in later versions of WebSphere Multichannel Bank Transformation Toolkit. In WebSphere Multichannel Bank Transformation Toolkit version 8.0, the `dse.ini` file has been reorganized and renamed as `btt.xml`. The migration tool helps you to migrate the `dse.ini` file to the `btt.xml` file.
- Migrate other XML files:

  In WebSphere Multichannel Bank Transformation Toolkit, contexts, formatters, services, data elements, and operations are externalized into XML files. Because the referenced package names and class names of these XML files are different in different versions of WebSphere Multichannel Bank Transformation Toolkit, the migration tool automatically changes the package names and class names.
- Generate migration report:

  After you run the migration tool, a report is generated to record the tasks that the migration tool performed. The migration report includes information on files that have been migrated by the tool and files that must be migrated manually.

Migration procedures

This topic provides an outline of the procedures that are involved in migrating IBM WebSphere Multichannel Bank Transformation Toolkit version 5.2 to WebSphere Multichannel Bank Transformation Toolkit version 8.0.

The migration process consists of the following four phases:

- Phase 1: Analysis and preparation

- Phase 2: Customizing the migration tool
- Phase 3: Migrating your applications
- Phase 4: Adding business logic and fixing errors

The phases are iterative and contain tasks that might be refined during the project life cycle.

For more information on each phase of the migration process, refer to the following topics.

Phase 1: Analysis and preparation

In phase 1, you must analyze the differences between IBM WebSphere Multichannel Bank Transformation Toolkit version 5.2 and WebSphere Multichannel Bank Transformation Toolkit version 8.0. Particular attention must be paid to the application logic layer.

About this task

You must identify the parts of WebSphere Multichannel Bank Transformation Toolkit version 5.2 that can be migrated automatically and the parts that cannot be migrated automatically, and then make necessary preparation for the migration.

Procedure

To prepare for the migration, perform the following tasks:

1. Gather the definition files of the existing application system version 5.2. Mark your extensions for future verification.
2. Prepare your application code and JSP sources. The migration tool migrates the Java code of the application to WebSphere Multichannel Bank Transformation Toolkit version 8.0.
3. Prepare to manually migrate the WebSphere Multichannel Bank Transformation Toolkit version 5.2 invoker. WebSphere Multichannel Bank Transformation Toolkit version 5.2 uses the invoker framework to invoke the business logic. WebSphere Multichannel Bank Transformation Toolkit version 5.2 leverages EJB as the business logic implementation. The invoker of WebSphere Multichannel Bank Transformation Toolkit version 5.2 is mainly used for EJB calling from the WebSphere Multichannel Bank Transformation Toolkit runtime server. In WebSphere Multichannel Bank Transformation Toolkit version 8.0, the request from the channel side will directly call the WebSphere Multichannel Bank Transformation Toolkit processor or operation in the WebSphere Multichannel Bank Transformation Toolkit runtime container.
4. WebSphere Multichannel Bank Transformation Toolkit version 8.0 provides more powerful context, and the API usage is different from that in WebSphere Multichannel Bank Transformation Toolkit version 5.2. The migration tool of WebSphere Multichannel Bank Transformation Toolkit version 8.0 can help you migrate the context and APIs.
5. Verify version 5.2 services. The supported services of WebSphere Multichannel Bank Transformation Toolkit version 8.0 on the server side support all of the WebSphere Multichannel Bank Transformation Toolkit version 5.2 services. The migration tool helps to migrate the package names, class names, APIs, and so on. If you have customized services, you can customize the migration rule to meet your migration requirements.
6. Verify your event usage. The event mechanism has been changed to fit the version 8.0 framework so that it can be distributed, and it spans different

servers. The event mechanism of version 8.0 is also compatible with the standard usage of version 5.2. You must perform some manual migration to run the WebSphere Multichannel Bank Transformation Toolkit version 5.2 events in WebSphere Multichannel Bank Transformation Toolkit version 8.0 if you had special usages.

7. Verify your client type. There are two kinds of client applications in WebSphere Multichannel Bank Transformation Toolkit version 5.2. There is no change to the Swing-based Javaclient, so you do not need to migrate it to WebSphere Multichannel Bank Transformation Toolkit version 8.0. For JSP/HTML-based clients, the migration tool migrates JSP Tablib uri, WebSphere Multichannel Bank Transformation Toolkit customized tags, page import, and Java code in JSP files.
8. Migrate the WebSphere Multichannel Bank Transformation Toolkit trace. WebSphere Multichannel Bank Transformation Toolkit version 8.0 trace is completely compatible with WebSphere Multichannel Bank Transformation Toolkit version 5.2 trace. If you do not want to use the new trace framework, the old trace can still work in the WebSphere Multichannel Bank Transformation Toolkit version 8.0 runtime container.

Phase 2: Customizing the migration tool

Phase 2 of the migration process involves the customization of the migration tool to extend the functions of the tool.

About this task

IBM WebSphere Multichannel Bank Transformation Toolkit is a framework and can be extended. Because the default rules of the migration tool are intended to migrate WebSphere Multichannel Bank Transformation Toolkit applications that have not been extended, you must extend the migration tool if you have extended WebSphere Multichannel Bank Transformation Toolkit.

Note: It is not mandatory for you to customize the migration tool if you have extended WebSphere Multichannel Bank Transformation Toolkit. If manually performing the migration of extended WebSphere Multichannel Bank Transformation Toolkit applications is more convenient than customizing the migration tool, then do not customize the migration tool.

Procedure

To customize the migration tool, perform the following steps:

1. Customize the Java code migration rules.
2. Customize the migration rules for the definition files. If you have customized tags in the definition files, you should customize the migration rules to meet these special requirements.

Phase 3: Migrating your applications

Phase 3 of the migration process involves the migration of your applications with the migration tool.

Procedure

To migrate your applications from WebSphere Multichannel Bank Transformation Toolkit version 5.2 to WebSphere Multichannel Bank Transformation Toolkit version 8.0, do the following steps:

1. Replace the WebSphere Multichannel Bank Transformation Toolkit version 5.2 JAR files with WebSphere Multichannel Bank Transformation Toolkit version 8.0 JAR files, and change the build path of the project to the new JAR files.
2. Import the application that you want to migrate to the workspace as a project, and configure the migration tool. The migration tool migrates the application directly based on the original project.
3. Migrate the `dse.ini` configuration file to the `btt.xml` configuration file. In WebSphere Multichannel Bank Transformation Toolkit version 5.2, the configuration file is `dse.ini`, but in WebSphere Multichannel Bank Transformation Toolkit version 8.0, the `dse.ini` configuration file has been renamed as `btt.xml`.
4. Migrate the definition files of global definition files and self-defined operation and processor.
5. Migrate the business logic. In WebSphere Multichannel Bank Transformation Toolkit version 8.0, the architecture is refined, and some components are re-designed; therefore, the package names, class names, and some APIs in WebSphere Multichannel Bank Transformation Toolkit version 8.0 are different from those in WebSphere Multichannel Bank Transformation Toolkit version 5.2. However, the functions are still compatible. The migration tool will automatically migrate these differences.
6. Manually migrate the components that cannot be completely migrated by the migration tool. Usually, failure to migrate components with the migration tool are caused by errors. For information how to fix the errors, refer to the “Phase 4: Adding the business logic and fixing errors” topic.

Phase 4: Adding the business logic and fixing errors

After you have migrated the applications from IBM WebSphere Multichannel Bank Transformation Toolkit version 5.2 to IBM WebSphere Multichannel Bank Transformation Toolkit version 8.0, you must add the business logic and fix the errors

Procedure

To add the business logic and fix errors, perform the following steps:

Fix the Java build errors. There might be API mismatch problems between WebSphere Multichannel Bank Transformation Toolkit version 5.2 and WebSphere Multichannel Bank Transformation Toolkit version 8.0. The migration tool can resolve most of the problems for manual migration. If the problems are not resolved by the migration tool, use Rational Application Developer to resolve the problems.

Migration tasks by migration tool

This section describes how to set up the migration tool and how to use the migration tool to perform migration tasks.

IBM WebSphere Multichannel Bank Transformation Toolkit version 8.0 provides a migration tool to help you migrate the applications that you developed with WebSphere Multichannel Bank Transformation Toolkit version 5.2 the version 8.0 architecture.

The migration tool can perform the following tasks to help you to migrate your applications:

- Convert the `dse.ini` file from WebSphere Multichannel Bank Transformation Toolkit version 5.2 to the `btt.xml` file in WebSphere Multichannel Bank Transformation Toolkit version 8.0.
- Migrate the definition files inWebSphere Multichannel Bank Transformation Toolkit version 5.2 to the definition files in WebSphere Multichannel Bank Transformation Toolkit version 8.0, including context definitions, formatter definitions, and so on.
- Migrate the application Java code, including package names, class names, APIs, and so on from WebSphere Multichannel Bank Transformation Toolkit version 5.2 to WebSphere Multichannel Bank Transformation Toolkit version 8.0.
- Migrate the JSP Tablib URI, WebSphere Multichannel Bank Transformation Toolkit customized tags, page import, and Java code in JSP file from WebSphere Multichannel Bank Transformation Toolkit version 5.2 to WebSphere Multichannel Bank Transformation Toolkit version 8.0.

For detailed information on how to set up the migration tool and on the tasks that are performed by the migration tool, refer to the following topics.

Setting up the migration tool

Before you can migrate your applications from IBM WebSphere Multichannel Bank Transformation Toolkit version 5.2 to WebSphere Multichannel Bank Transformation Toolkit version 8.0, you must set up the migration tool.

Before you begin

If you installed WebSphere Multichannel Bank Transformation Toolkit before you installed IBM Rational Application Developer on your workstation, you must copy the following plugin files to the `$D(RAD)/plugins` directory manually after you install Rational Application Developer:

- **com.ibm.btt.tools.migration.0**
- **com.ibm.btt.core.0**

Procedure

To set up the migration tool, take the following steps:

1. Start Rational Application Developer.
2. Import the project that you want to migrate.
3. In Rational Application Developer, click **Window** > **Preferences**.
4. In the Preferences window, expand the **BTT** folder, and then click **Migration**.
5. In the Migration page, select the migration rule definition files that you want to apply. In the migration tool plug-in, there is a **config** category, and you can find WebSphere Multichannel Bank Transformation Toolkit version 5.2 to WebSphere Multichannel Bank Transformation Toolkit version 8.0 migration rule definition files under the related category.
6. Click **OK**.
7. In the Project Explorer view, right-click the BTT definition file and select **BTT Migration**.

Results

The migration report and migration results will be displayed in the Project Explorer view if there are any.

Adding rules

About this task

The migration tool enables different types of migration: Java migration, WebSphere Multichannel Bank Transformation Toolkit definition XML file migration, `dse.ini` file migration, and JSP file migration. The migration is implemented according to the configured migration rules.

Two sets of migration rules are provided: version 4.3 to version 8.0 migration rule set and version 5.2 to version 8.0 migration rule set. The rule sets are in the `$D(RAD)\plugins\com.ibm.btt.tools.migration.0\config` directory. By default, version 5.2 to version 8.0 migration rule set is applied.

The following files are configuration files for the rule-based Migration tool:
- `javaRule.xml`: rules for Java code migration
- `cfgRule.xml`: rules for XML and `dse.ini` file migration
- `bttTemplate.xml`: `btt.xml` template.
- `jspRule.xml`: rules for JSP file migration

If the default rules do not meet your migration requirements, you can add new rules and configure them as the migration rules.

Procedure

To configure your own migration rules, do the following steps:
1. In IBM Rational Application Developer, click **Window** > **Preferences**
2. In the Preferences window, expand the **BTT** folder, and then select **Migration**. The Migration page displays in the Preferences window.
3. In the Migration page, select the migration configuration files that you require.
   - If the default migration rules that are provided by the WebSphere Multichannel Bank Transformation Toolkit migration tool meet your migration requirements, select the default migration rules.
   - If the default migration rules that are provided by the WebSphere Multichannel Bank Transformation Toolkit migration tool do not meet your migration requirements, define your own migration rules, and then click **Browse** to locate your migration configuration file.

   Note: If you do not specify any migration rules in the Migration page, the default migration rules will be used during migration.

Migrating the dse.ini file

This section describes how to migrate the `dse.ini` file.

Procedure

In IBM Rational Application Developer, right-click the `dse.ini` file, and then click **BTT Migration** > **Migrate dse.ini**.

Results

The `dse.ini` file is migrated to the **config.migrated** folder.

Migrating XML file

This section describes how to migrate an XML file.

Procedure

In the Enterprise Explorer pane of IBM Rational Application Developer, right-click the XML file that you want to migrate, and then click **BTT Migration** > **Migrate XML File**.

Note: If you want to migrate multiple XML files at the same time, press Shift and select the XML files that you want to migrate, and then right-click the selected XML files.

Results

The XML files are migrated to the **config.migrated** folder. If you selected multiple XML files for migration, all the XML files that you selected are migrated to the **config.migrated** folder at the same time.

Migrating Java code

About this task

When you migrate Java code, you can select to migrate either individual Java files or packages.

Procedure

To migrate Java code, do the following steps:

In the Enterprise Explorer pane of IBM Rational Application Developer, right-click the Java file or package that you want to migrate, and then click **BTT Migration** > **Migrate Java Files**.

Results

The Java code is migrated.

Migrating Context

This topic describes how to migrate a IBM WebSphere Multichannel Bank Transformation Toolkit version 5.2 context to a WebSphere Multichannel Bank Transformation Toolkit version 8.0 context.

About this task

You do not need to change the context, type, and data definitions of WebSphere Multichannel Bank Transformation Toolkit version 5.2 when you migrate them to WebSphere Multichannel Bank Transformation Toolkit version 8.0, because these definitions are the same in version 5.2 and version 8.0. In WebSphere Multichannel Bank Transformation Toolkit version 8.0, a class table tag for context implementation class and an initializer tag are added in the CHA configuration in the `btt.xml` definition file.

The APIs in WebSphere Multichannel Bank Transformation Toolkit version 5.2 and WebSphere Multichannel Bank Transformation Toolkit version 8.0 are almost the same except the way of constructing the context instance. In WebSphere Multichannel Bank Transformation Toolkit version 8.0, the context is constructed from `ContextFactory`. The WebSphere Multichannel Bank Transformation Toolkit version 5.2 CHA `startMode` definition tag (value =`"PersistenceShared"` or `"MemoryShared"`) is replaced by the `isPersistenceEnabled` tag (value=`"yes"` or `"no"`). An example of migrating a WebSphere Multichannel Bank Transformation Toolkit version 5.2 context to a WebSphere Multichannel Bank Transformation Toolkit version 8.0 context is provided in this topic.

Procedure

1. Creating a WebSphere Multichannel Bank Transformation Toolkit version 5.2context.

   ```
   Context ctxt = (Context) Context.readObject("myContext",true);
   Context ctxt2 = new com.ibm.dse.base.Context();
   ctxt2.setName("myContext2");
   ```

2. Creating a WebSphere Multichannel Bank Transformation Toolkit version 8.0 context.

   ```
   Context diiTestCtx = ContextFactory.createContext("myContext", true);
   Context ctxt2 = new com.ibm.btt.base.Context();
   ctxt2.setName("myContext2");
   ```

Example

The following is an example of a WebSphere Multichannel Bank Transformation Toolkit version 8.0 CHA configuration. Please note the bold text that is different from WebSphere Multichannel Bank Transformation Toolkit version 5.2 CHA configuration.

```
<kColl id="cha">
    <!-- indicates the implementation class of context-->
    <kColl id="classTable">
      <field id="context" value="com.ibm.btt.cha.ejb.RemoteContextImpl" />
    </kColl>

    <!-- indicates the initialization class of CHA-->
    <field id="initializer" value="com.ibm.btt.base.ContextInitializer" />
    <!-- indicates the external file of CHA-->
    <field id="extFile" value="dsectxt.xml" />

    <!-- indicates the JNDI looking up properties of CHA-->
    <field id="ejbInitialContextFactory"
      value="com.ibm.websphere.naming.WsnInitialContextFactory" />
    <field id="EJBProviderURL" value="iiop://localhost:2809" />

    <!-- indicates the home interface of CHAEJB-->
    <field id="CHASessionLocalHome"
      value="java:comp/env/ejb/CHASession" />
    <field id="CHAInstanceLocalHome"
    value="java:comp/env/ejb/CHAInstance" />
    <field id="CHAControlLocalHome"
    value="java:comp/env/ejb/CHAControl" />
    <field id="CHASessionHome"
    value="ejb/com/ibm/btt/cha/ejb/CHASessionHome" />
    <field id="CHAInstanceHome"
    value="ejb/com/ibm/btt/cha/ejb/CHAInstanceHome" />

    <!-- indicates the initialized context tree of CHA-->
    <field id="initTailContextName" value="branchServer" />

    <!-- indicates whether to cleanup the context instance from the DB tables-->
    <field id="cleanupCHAServer" value="true" />
```

```
    <!-- indicates whether to use local interface when invoke CHAEJB-->
    <field id="isLocalCall" value="true" />

    <!-- indicates whether the remote CHA supports persistence-->
    <field id="isPersistenceEnabled" value="false" />
    <!--field id="startMode" value="MemoryShared"/-->
</kColl>
```

Migrating Formatter

About this task

To migrate a formatter, you must migrate the configuration that is related to the formatter in the `dse.ini` file in IBM WebSphere Multichannel Bank Transformation Toolkit version 5.2 to the `btt.xml` file in WebSphere Multichannel Bank Transformation Toolkit version 8.0. Do not change the other code in the file.

Migrating Session Management

This topic describes how to migrate a IBM WebSphere Multichannel Bank Transformation Toolkit version 5.2 Session management to a WebSphere Multichannel Bank Transformation Toolkit version 8.0 Session management.

About this task

WebSphere Multichannel Bank Transformation Toolkit provides the `Public static void markSessionExpired(HttpSession aSession) throws BTTSMException` utility method in the `com.ibm.btt.sm.CSSessionHandler` class. After this method is called, the session is marked as expired and is removed at the end of operation. You need to call this method in the logoff operation. Then, the session context will be cleaned and the HTTP session object will be destroyed after you log off the application.

WebSphere Multichannel Bank Transformation Toolkit version 5.2 supports both the `cookie=true` and the `cookie=false` modes.

- When `cookie=true`, the HttpSessionHandler object is used in the HttpSession that implements the `HttpSessionBindingListener` interface. When the HttpSession is invalidated, the `endSession(sessionId)` method is triggered.
- When `cookie=false`, the WebSphere Multichannel Bank Transformation Toolkit channel is responsible for the session expiration. It uses a dedicated thread, `CSSessionManager`, to check the expired sessions regularly with the `checkExpiredSessions()` method. And it ends the expired WebSphere Multichannel Bank Transformation Toolkit session.

In WebSphere Multichannel Bank Transformation Toolkit version 8.0, only the `cookie=true` mode is supported. Session management is delegated to HttpSession. You must add the HttpSession listener in your WAR project for session management.

To migrate the `cookie=false` mode in WebSphere Multichannel Bank Transformation Toolkit version 5.2, you must migrate your old mechanism to the HttpSession timeout mechanism. Add the following HttpSession timeout listener in the `web.xml` file:

```
<listener>
<listener-class>com.ibm.btt.sm.TimeoutHandler</listener-class>
</listener>
```

Migrating Channel

About this task

The channel component is restructured in IBM WebSphere Multichannel Bank Transformation Toolkit version 8.0. Besides `ChannelRequest` and `ChannelResponse`, which help to achieve protocol and channel independency, there are some other changes in structure and code, including:

- In WebSphere Multichannel Bank Transformation Toolkit version 5.2, `CSReqServlet` implements `ChannelDriver`. In BTT version 8.0, `CSReqServlet` is only a servlet and does not implement the `ChannelDriver` interface.
- In WebSphere Multichannel Bank Transformation Toolkit version 7.0, the `BTTChannelDriver` class was introduced, which implements the `ChannelDriver` interface. In WebSphere Multichannel Bank Transformation Toolkit version 8.0, the `BTTChannelDriver` class is still used to implement the `ChannelDriver` interface.
- Because `BTTChannelDriver` implements `ChannelDriver` in WebSphere Multichannel Bank Transformation Toolkit version 8.0, some functions and methods in `CSReqServlet` are not in `ChannelDriver`.

To migrate the customer extended channel request handler and response handler, you only need to refactor your application to the WebSphere Multichannel Bank Transformation Toolkit version 8.0 channel interface and move some methods and logic to the classes in WebSphere Multichannel Bank Transformation Toolkit version 8.0.

Migrating exception handling

About this task

In IBM WebSphere Multichannel Bank Transformation Toolkit version 5.2, APIs such as `context.getKeyedCollection()`, `context.getValue()` do not throw out exceptions; while in WebSphere Multichannel Bank Transformation Toolkit version 8.0, these APIs throw exceptions.

As a result, you must migrate exception handling and trace the exceptions.

The following code is the code sample before migration:

```
return utb.getContext().getKeyedCollection().getElements().entrySet();
```

After migration, the code is as follows:

```
try {
return utb.getContext().getKeyedCollection().getElements().entrySet();
} catch (DSEInvalidRequestException e) {
// TODO: Exception handling
//Trace the exception.
}
return null;
```

Migrating externalizer

About this task

To migrate the externalizer, you must migrate the `getSetting()` parameter and the iniFile path.

- Migrating the `getSetting()` parameter:

In WebSphere Multichannel Bank Transformation Toolkit version 8.0, each component has its own initializer; therefore, you need the initializer of each component to retrieve the parameter.

Following is the code sample before migration:

```
String startupOpName = (String) Settings.getSettings().getValueAt("channelHandlers."
+ channelContext.getDeviceType() + ".startupOp");
```

After you migrate to WebSphere Multichannel Bank Transformation Toolkit version 8.0, the code is as follows:

```
String startupOpName = (String) ChannelInitializer.getSettings().
getValueAt(channelContext.getDeviceType() + ".startupOp");
```

- Migrating iniFile path:

  WebSphere Multichannel Bank Transformation Toolkit version 8.0 supports the following types of iniFile:

  - JAR path iniFile, such as `jar:///package/btt.xml`
  - File path iniFile, such as `file:///path/btt.xml`
  - URL path iniFile, such as `http://127.0.0.1:9080/BankWeb/path/btt.xml`

  If you use the file path way, you need to migrate the iniFile in the `StartServerServlet` class:

```
String res=null;
if (!iniFileParameter.startsWith("/WEB-INF")) {
File iniFile=new File(iniFileParameter);
if (iniFile.isAbsolute() && iniFile.exists()) {
res=iniFile.getAbsolutePath();
}
res = "file:\\\\\\" + res;
}
```

Migrating service

About this task

IBM WebSphere Multichannel Bank Transformation Toolkit version 8.0 supports the definition of a service in a context. Following is the code example in the `service.xml` file:

```
<CSServer id="realCSServer" inactivityClientTimeout="900000"
timeBetweenSessionCheck="60000"/>
```

The code example in the `context.xml` file is as follows:

```
<context id="rootCtx" type="root">
<refKColl refId="rootData"/>
<refService refId="realCSServer" alias="CSServer" type="cs"/>
<refService refId="JDBCIB2ConnectionManager" alias="connectionManager"
type="jdbcManager"/>
<refService alias="proximaService" refId="proximaService" type="prx"/>
<refService alias="dummyService" refId="dummyService" type="prx"/>
<refService alias="messageResourcesSrv" refId="realMessageResourcesService"
type="multilan"/>
</context>
```

And the following code is the Java code before migration:

```
Service.readObject(getCSServerServiceName());
```

After migration, the Java code is as follows:

```
((CSServerService)ContextFactory.getCSServer())
```

Migrating trace

This topic describes how to migrate the IBM WebSphere Multichannel Bank Transformation Toolkit version 5.2 trace to the WebSphere Multichannel Bank Transformation Toolkit version 8.0 trace.

About this task

The original trace API of WebSphere Multichannel Bank Transformation Toolkit version 5.2 can be still used in WebSphere Multichannel Bank Transformation Toolkit version 8.0. The WebSphere Multichannel Bank Transformation Toolkit version 8.0 trace facility can automatically map the old API to theAPI of the WebSphere Multichannel Bank Transformation Toolkit version 8.0 trace facility. WebSphere Multichannel Bank Transformation Toolkit version 8.0 traces by package, but the original trace is by the Component ID in WebSphere Multichannel Bank Transformation Toolkit version 5.2. The WebSphere Multichannel Bank Transformation Toolkit version 5.2 trace can trace to multiple targets at the same time, but WebSphere Multichannel Bank Transformation Toolkit version 8.0 supports tracing to one target at the same time only when the `BTTLogFactoryImplementClass` class is set.

The implementation of trace to File is changed to using Java util API, so the original parameter of trace to File is not supported in WebSphere Multichannel Bank Transformation Toolkit version 8.0. The original function and implementation of trace to Display is kept in WebSphere Multichannel Bank Transformation Toolkit version 8.0, so the parameter for trace to Display is still supported.

The original HML trace level is not supported in WebSphere Multichannel Bank Transformation Toolkit version 8.0. The original trace type is mapped to a new trace level. See the following table.

Table 1. Trace level map

WebSphere Multichannel Bank Transformation Toolkit version 8.0 trace level	WebSphere Multichannel Bank Transformation Toolkit version 5.2 trace type	WAS trace level	Common logging	Log4J
FATAL	Severe	Fatal	Fatal	Fatal
ERROR	Error	Severe	Error	Error
WARN	Warning	Warning	Warn	Warn
INFO	Information Display	Info	Info	Info
DEBUG	Debug AllTypes	Detail*	Debug	Debug

Example

An example of migrating WebSphere Multichannel Bank Transformation Toolkit version 5.2 trace to WebSphere Multichannel Bank Transformation Toolkit version 8.0 trace is as follows:

WebSphere Multichannel Bank Transformation Toolkit version 5.2 trace configuration

```
<kColl id="traces">
    <field id="initializer" value="com.ibm.btt.base.TraceInitializer" />
    <field id="traceToFile" value="yes" />
    <field id="traceFileName" value="c:\btt\log\btt.log" />
    <field id="traceToDisplay" value="no" />
    <field id="traceToWAS" value="yes" />
    <field id="traceWindowTitle" value="Server Trace" />
    <field id="showOriginator" value="yes" />
    <field id="showWarningMessage" value="no" />
    <field id="traceLevels" value="debug" />
    <field id="traceMaxLogFiles" value="5" />
    <field id="font" value="monospaced" />
    <field id="createBackup" value="yes" />
    <field id="fileNumberOfLines" value="4000" />
    <field id="displayNumberOfLines" value="2000" />
    <field id="linesOfBuffer" value="7000" />
    <field id="lineLength" value="200" />

    <kColl id="requestersComponents">
      <traceRequester id="#CHA" trace="yes" traceLevels="HML" traceTypes="FATAL" />
      <traceRequester id="#CS" trace="yes" traceLevels="HML" traceTypes="DEBUG" />
    </kColl>
</kColl>
```

WebSphere Multichannel Bank Transformation Toolkit version 8.0 trace configuration after migration

```
<kColl id="traces">
    <field id="initializer" value="com.ibm.btt.base.TraceInitializer" />
    <field id="traceTargetFactoryImplClass"
  value="com.ibm.btt.base.BTTLogFactoryToDisplayImp" />
    <field id="displayNumberOfLines" value="2000" />
    <kColl id="requestersComponents">
      <traceRequester id="com.ibm.btt.base.*" trace="yes" traceLevels="FATAL" />
      <traceRequester id="com.ibm.btt.channel.*" trace="yes" traceLevels="DEBUG" />
    </kColl>
</kColl>
```

WebSphere Multichannel Bank Transformation Toolkit version 5.2 trace application code

```
if (Trace.doTrace(Constants.CHACOMPID,Trace.High,Trace.Debug))
    Trace.trace(Constants.CHACOMPID,Trace.High,Trace.Debug,Settings.getTID(),
              " CHA Debug .........");
if (Trace.doTrace(Constants.CHACOMPID,Trace.High,Trace.Information))
    Trace.trace(Constants.CHACOMPID,Trace.High,Trace.Information,Settings.getTID(),
             "CHA info .........");
```

WebSphere Multichannel Bank Transformation Toolkit version 8.0 trace application code after migration

```
BTTLog   log=BTTLogFactory.getLog("com.ibm.btt.base.LocalContextImp");
If (log.doDebug())
  log.debug("CHA Debug......");
If (log.doInfo())
  log.info("CHA info......");
```

Four trace types

WebSphere Multichannel Bank Transformation Toolkit version 8.0 supports the following four trace target types:

- Trace to window
- Trace to self define file
- Trace to WAS file

- Trace to common-logging
- Trace to window and self-defined file:

  Trace to window and trace to self-defined file functions are kept for migration consideration.

  Trace to window function can be used at client side.

  Most of the functions of trace to file are not supported in WebSphere Multichannel Bank Transformation Toolkit version 8.0. For example, the following is not supported:

  ```
  <field id="traceMaxLogFiles" value="5"/>
  <field id="fileNumberOfLines" value="4000"/>
  <field id="displayNumberOfLines" value="200"/>
  <field id="linesOfBuffer" value="700"/>
  <field id="lineLength" value="128"/>
  ```

  As a result, you can use trace to common logging or trace to WAS instead.
- Trace to WAS and common logging:

  WebSphere Multichannel Bank Transformation Toolkit version 8.0 supports trace to WAS and common logging. WebSphere Multichannel Bank Transformation Toolkit trace maps the WebSphere Multichannel Bank Transformation Toolkit trace API to the corresponding trace API of common logging and WAS java.util.logging.

  Note: Common logging only provides the API standard, and the logging implementation is provided by other logging facilities for example Log4J. The trace level is also converted.

  1. Trace to WAS:
     a. Change the `dse.ini` file, and enable traceToWAS. You can enable and disable trace switch and set trace level for each component.
     b. You can enable and change the trace level dynamically in WAS console.
     c. Trace application code calls the WebSphere Multichannel Bank Transformation Toolkit trace API. You can enter component ID and trace level as parameters.
     d. The WebSphere Multichannel Bank Transformation Toolkit trace facility calls the corresponding `java.util.logging` API, if you enable the component and WebSphere Multichannel Bank Transformation Toolkit trace level. The WebSphere Multichannel Bank Transformation Toolkit trace level is converted to WAS trace level.
     e. After WebSphere Multichannel Bank Transformation Toolkit trace calls the WAS trace API, and if the input trace level is enabled in the trace configuration of WAS console, the trace content is recorded to WAS trace file.
  2. Trace to common logging:
     a. Change the `dse.ini file` and enable traceToCommonLogging.
     b. You can enable or disable the trace switch and set the trace level for each component.
     c. Set the common logging property file and specify the implementation of common logging, for example Log4J.
     d. Trace application code calls the corresponding common logging API. You can enter component ID and trace level as parameters.
     e. WebSphere Multichannel Bank Transformation Toolkit trace facility calls the corresponding common-logging API, if you enable the component and the WebSphere Multichannel Bank Transformation Toolkit trace level.

The WebSphere Multichannel Bank Transformation Toolkit trace level is converted to common logging trace level.

f. WebSphere Multichannel Bank Transformation Toolkit trace calls the common logging API. Whether the trace content is recorded into the trace file depends on the trace level setting of the common logging implementation.

3. Migrating code:
   - Migrating XML definition files.

     Before migration, the code for XML definition file is as follows:

```
<kColl id="traces">
  <field id="traceToFile" value="yes"/>
  <field id="traceToDisplay" value="yes"/>
  <field id="traceFileName" value="c:\dse\log\btt.log"/>
  <field id="traceMaxLogFiles" value="100"/>
  <field id="font" value="monospaced"/>
  <field id="createBackup" value="yes"/>
  <field id="fileNumberOfLines" value="4000"/>
  <field id="displayNumberOfLines" value="200"/>
  <field id="linesOfBuffer" value="700"/>
  <field id="lineLength" value="128"/>
  <field id="showOriginator" value="yes"/>
  <field id="useServletsEngineLog" value="no"/>
  <field id="servletsEngineLogPort" value="80"/>
  <field id="showWarningMessage" value="yes"/>
  <field id="traceWindowTitle" value="PSP6 Server Trace"/>
  <field id="traceTypes" value="DIPEWSV"/>
  <field id="traceLevels" value="HML"/>
  <field id="showContextDump" value="yes"/>
  <kColl id="requestersComponents">
  ....
  </kColl>
</kColl>
```

     After migration, the code is as follows:

```
<kColl id="traces">
  <field id="initializer" value="com.ibm.btt.base.TraceInitializer"/>
  <field id="traceTargetFactoryImplClass"
   value="com.ibm.btt.base.BTTLogFactoryToCommonLoggingImp" />
  <field id="showContextDump" value="yes"/>
  ....
</kColl>
```

     In the XML trace definition migration, `<requestersComponents>` definition is not used for common logging. You need to configure `initializer` and `traceTargetFactoryImplClass`. You can leave the other configuration as it is.

     If you choose common logging trace, you must also migrate the `TraceConfigServlet` class.

     If you choose WAS trace, WAS console has similar functions.
   - Some trace APIs do not need migration, for example:

```
Trace.trace(CommonsConstants.COMPID, Trace.Low,
Trace.Information, Settings.getTID(), msg);
Trace level=Trace.Low, Setting.getTID() will not use,
BTT trace will map the other three parameter to commond-logging.
```

Migrating JSP files

About this task

You can migrate JSP Tablib uri, WebSphere Multichannel Bank Transformation Toolkit customized tags, page import, and Java code in JSP files. The migration rules of taglib uri and WebSphere Multichannel Bank Transformation Toolkit customized tags are defined in the JSP migration rule file. The migration rules of Java code are defined in the Java migration rule file.

To migrate JSP file, perform the following steps:

Procedure

To migrate a JSP file, do the following steps:

1. In the Enterprise Explorer pane of IBM Rational Application Developer, right-click the JSP file that you want to migrate. If you want to migrate multiple JSP files at the same time, press Shift and select the JSP files that you want to migrate.
2. Click **BTT Migration** > **Migrate JSP file**.
3. The JSP file is then migrated to the jsp.migrated folder.

Results

The JSP file is migrated to the **jsp.migrated** folder. If you selected multiple JSP files for migration, all the JSP files are migrated to the **jsp.migrated** folder at the same time.

The following tables describe the samples for each migration type:

Table 2. Taglib uri migration

Before migration	Migration rule	After migration
<%@ taglib uri="/WEB-INF/dse.tld" prefix="dse" %>	<taglibRule oldTagUrl="/WEB-INF/dse.tld" newTagUrl="/WEB-INF/btt.tld" oldTagPrefix="dse" newTagPrefix="btt"/>	<%@ taglib uri="/WEB-INF/btt.tld" prefix="btt" %>

Table 3. Tag migration

Before migration	Migration rule	After migration
<dse:label text="jspMigrationToolTest"/>	<tagRule oldTitle="dse:label" newTitle="btt:label" oldKey="text" newKey="text_new" oldVal="jspMigrationToolTest" newVal= "jspMigrationToolTest_new" />	<btt:label text_new= "jspMigrationToolTest_new" />

Table 4. Page import migration

Before migration	Migration rule	After migration
<%@page import="com.ibm.dse" %>	<pageImport oldPattern="com.ibm.dse" newPattern="com.ibm.btt"/>	<%@page import=“com.ibm.btt"%>

Table 5. Java code migration

Before migration	Migration rule	After migration

com.ibm.dse.base.
JavaExtensions.
getAlphaUniqueCode()
	<simpleRule
	oldItem="*com.ibm.dse*"
	newItem="*com.ibm.btt*" />
		com.ibm.btt.base.
		JavaExtensions.
		getAlphaUniqueCode()

Migrating APIs

About this task

Errors might be encountered when migrating APIs because some API names are duplicated. To ensure that APIs are migrated, perform the following changes:

Procedure

- Change `Settings.reset(iniFile)` to `InitManager.reset(iniFile)`
- Use `Context ctx=(Context)ContextFactory.createContext(name, false)` to create context, where `False` refers to local context.
- Change `(HttpServletRequest)channelContext.getChannelRequest()` to `(HttpServletRequest)channelContext.getChannelRequest().getRequest()`.
- Change `(HttpServletResponse)channelContext.getChannelResponse()` to `(HttpServletResponse)channelContext. getChannelResponse().getResponse()`.
- Change `Context.getRoot()` to `ContextFactory.getRoot()`.
- Change `formatter.unformat(Label13Value,Operation)` to `formatter.unformat(Label13Value,Context)`.
- Change `Formatter.format(Operation)` to `Formatter.format(Context)`.
- In the HtmlRequestHandler, change return type to `Object` in `public ServerOperation executeRequest(ChannelContext channelContext)`.

Generating reports

About this task

Migration reports are generated after migration that identify the tasks that were completed during the migration process. Separate migration reports are generated for Java code migration, JSP file migration and WebSphere Multichannel Bank Transformation Toolkit XML definition file migration.

Note: The migration report files will not be generated for the migration of the `dse.ini` file.

All the report files are generated in a folder named report. The `changehistory_*.txt` file records all of the items that have been migrated during the migration process. The `error_*.txt` file provides a record of all of the errors that occurred during the migration process. The `summary_*.txt` file provides a summary of all of the migrated files and its status.

Manual migration tasks

Some migration tasks cannot be performed automatically with the migration tool because the migration tool can only perform defined rules; therefore, you must perform some migration tasks manually. For reference information that is related to the manual migration of IBM WebSphere Multichannel Bank Transformation

Toolkit version 5.2 to IBM WebSphere Multichannel Bank Transformation Toolkit version 8.0, refer to the topics in this section.

Manual migration

The migration tool cannot perform all the tasks that are involved in migrating from IBM WebSphere Multichannel Bank Transformation Toolkit version 5.2 to IBM WebSphere Multichannel Bank Transformation Toolkit version 8.0. You must perform the migration of the following items manually:

- Some new features in WebSphere Multichannel Bank Transformation Toolkit version 8.0, such as Web 2.0, Rich Client, and so on. The base functionality of the migration tool only migrates the existing applications to WebSphere Multichannel Bank Transformation Toolkit version 8.0 and checks that they can run in WebSphere Multichannel Bank Transformation Toolkit version 8.0. You must redesign your applications and implement them manually.
- The existing services that are out of the scope of WebSphere Multichannel Bank Transformation Toolkit version 8.0. The services that are in the WebSphere Multichannel Bank Transformation Toolkit version 5.2 application system are not included in the migration tool; therefore, you migrate them manually.
- The server-side event mechanism of WebSphere Multichannel Bank Transformation Toolkit version 8.0 has changed. You can define event related migration rules in WebSphere Multichannel Bank Transformation Toolkit version 8.0, but for complicated scenarios that migration rules cannot handle, you must migrate them manually.
- The communication service access uses the JCA connector in WebSphere Multichannel Bank Transformation Toolkit version 8.0. You must modify it manually.
- The WebSphere Multichannel Bank Transformation Toolkit version 5.2 invoker. The invoker is a hard coded EJB invocation framework. WebSphere Multichannel Bank Transformation Toolkit version 8.0 provides a common invocation framework for POJO, EJB, Web services, and JMS. These two frameworks are entirely different. The API usage and configuration are also entirely different. The EJB invocation information that is defined in the resource bundle file can be mapped to the EJB invoker definition in the file invoker.xml. The WebSphere Multichannel Bank Transformation Toolkit version 8.0 invoker is decoupled with WebSphere Multichannel Bank Transformation Toolkit context and formatter. The usage of WebSphere Multichannel Bank Transformation Toolkit version 8.0 invoker is simple and flexible compared with the usage of the WebSphere Multichannel Bank Transformation Toolkit version 5.2 invoker. For detailed information on the WebSphere Multichannel Bank Transformation Toolkit version 8.0, see the Invoker topic.

Code changes in dse.ini after migration

This section lists the code changes in the `dse.ini` after you migrate your application from IBM WebSphere Multichannel Bank Transformation Toolkit version 5.2 to IBM WebSphere Multichannel Bank Transformation Toolkit version 8.0.

You can use the migration tool to migrate the dse.ini file. After migration, the file name is changed from `dse.ini` to `btt.xml`.

Initializer and extFile

In IBM WebSphere Multichannel Bank Transformation Toolkit version 8.0, the configuration of each component is defined in the kColl, and each component has its own initializer.

In the following example, you can see that after the migration, each data's definition kColl has extFile and initializer:

Before migration, the code is as follows:

```
<kColl id="data">
<field id="field" value="com.ibm.btt.base.DataField"/>
<field id="iColl" value="com.fortis.be.rbaa.commons.data.IndexedCollection"
 description="compound"/>
<field id="kColl" value="com.ibm.btt.base.KeyedCollection" description="compound"/>
<field id="operDef" value="com.ibm.btt.base.OperField"/>
<field id="refData"/>
<field id="sessionEntry" value="com.ibm.btt.base.SessionEntry"/>
<field id="sessionTable" value="com.ibm.btt.base.SessionTable"/>
<!-- Fortis WSBCC Commons -->
<field id="finalField" value="com.fortis.be.commons.data.FinalDataField"/>
</kColl>
```

After migration, the code is changed to the following:

```
<kColl id="data">
<field id="extFile" value="dsedata.xml" />
<field id="initializer"
value="com.ibm.btt.base.DataInitializer" /><kColl id="classTable">
<field id="field" value="com.ibm.btt.base.DataField"/>
<field id="iColl" value="com.fortis.be.rbaa.commons.data.IndexedCollection"
 description="compound"/>
<field id="kColl" value="com.ibm.btt.base.KeyedCollection" description="compound"/>
<field id="operDef" value="com.ibm.btt.base.OperField"/>
<field id="refData"/>
<field id="sessionEntry" value="com.ibm.btt.base.SessionEntry"/>
<field id="sessionTable" value="com.ibm.btt.base.SessionTable"/>
<!-- Fortis WSBCC Commons -->
<field id="finalField" value="com.fortis.be.commons.data.FinalDataField"/>
</kColl>
</kColl>
```

Processor definition

About this task

In IBM WebSphere Multichannel Bank Transformation Toolkit version 5.2, the processor definition is as follows:

```
<kColl id="processors">
    <procDef id=".." value=".." path=".."/>
    ....
</kColl>
```

After you migrate to WebSphere Multichannel Bank Transformation Toolkit version 8.0, the processor definition is changed to the following:

```
<kColl id="processors">
    <kColl id="files">
        <procDef id=".." value=".." path=".."/>
        ....
    </kColl>
    ....
</kColl>
```

Operation definition

About this task

In IBM WebSphere Multichannel Bank Transformation Toolkit version 5.2, the operation definition is as follows:

```
<kColl id="operation">
  <procDef id=".." value=".." path=".."/>
  ....
</kColl>
```

After you migrate to WebSphere Multichannel Bank Transformation Toolkit version 8.0, the operation definition is changed to the following:

```
<kColl id="operation">
  <kColl id="files">
    <operDef id=".." value=".." path=".."/>
    ....
  </kColl>
  ....
</kColl>
```

Notices

IBM may not offer the products, services, or features discussed in this document in all countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Lab Director
IBM China Software Development Lab
Diamond Building, ZhongGuanCun Software Park, Dongbeiwang West Road No.8,
ShangDi, Haidian District, Beijing 100193 P. R. China

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples may include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java is a trademark of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.