Oracle® Siebel Retail Finance

Banking Application Developer’s Reference Guide

Release 8.1.1 for Siebel Branch Teller

E20653-01

March 2011

Siebel Retail Finance Banking Application Developer's Reference Guide, Release 8.1.1

E20653-01

Contents

Preface

This guide contains information about the application components, build environment, and the build process for Siebel Branch Teller, Version 8.1.1.

Audience

This document is intended for developers and engineers who work with the Siebel Branch Teller application and for administrators who install and configure this product.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at `http://www.oracle.com/accessibility/`.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/support/contact.html` or visit `http://www.oracle.com/accessibility/support.html` if you are hearing impaired.

Related Documents

For more information, see the following documents on Siebel Bookshelf on Oracle Technology Network (OTN):

- *Siebel Branch Teller Online Documentation Library Release 8.1.1*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

What's New in This Release

The following table lists the changes in this version of the documentation to support version 8.1.1 of the software.

What's New in Siebel Branch Teller Banking Application Developer's

Reference Guide, Version 8.1.1.

Topic	Description
Section 3.3, "ANT Installation"	This is a new section about the installation of ANT, in the chapter, Preparing the Build Environment.
Section 3.8, "Downloading and Configuring Log4J"	This is a new section about downloading and configuring Log4J, in the chapter, Preparing the Build Environment.
Section 3.9, "The Java Doc Design Documentation"	This is a new section about Java Doc design documentation, in the chapter, Preparing the Build Environment.
Section 3.10, "How to update the BankframeConstants file"	This is a new section about editing and updating BankframeConstants.properties file, in the chapter, Preparing the Build Environment.
Section 3.11, "Encrypting and decrypting"	This is a new section about encrypting the passwords of existing users and encrypting the user credentials for BI Publisher, in the chapter, Preparing the Build Environment.
Section 3.12, "Printing Framework"	This is a new section about Printing Framework using BI Publisher, in the chapter, Preparing the Build Environment.
Section 3.13, "Bringing the application to offline without shutting down the central server"	This is a new section about bringing the application to offline, without shutting down the central server, in the chapter, Preparing the Build Environment.
Chapter 4, "Database Updates"	This is a new chapter about the updates related to the databases, Oracle and DB2, in Release 8.1.1.

2 Banking Application Components

This chapter introduces Oracle's Siebel Branch Teller Banking Application components, and the process for rebuilding customized modules.

2.1 Banking Application Components

The Siebel Branch Teller Application includes the following components:

- Server-side business logic contained within the Enterprise Archive (EAR) file that deploys the Banking Applications as a set of enterprise JAR files.
- Java Swing-based front end for the Branch Teller client and the administration client deployed within the EAR file as Java WebStart-enabled applications.
- Branch Offline Server to support a subset of transactions should the Central Server go offline. The Branch Offline Server is comprised of a lightweight RMI application and a data store.
- Web application for MCA Services administration functionality.

3 Preparing the Build Environment

This chapter covers preparing the build environment and includes the following topics:

Note: Refer to the *Siebel Retail Finance System Requirements and Supported Platforms* document on Oracle Technology Network for information regarding the supported environments, including the supported database and application server versions.

3.1 Extracting the Source Code

Before you perform the build process, you must prepare the build environment. This preparation involves extracting the source code, support files, and build files from the versions of the Siebel Retail Branch Teller Software Resources and product Build Packs that are appropriate to the product or platform for which they are being built. The build process is currently supported on the Windows platform only. As part of the installation process, the Banking Application pack is extracted to d:\siebel. See the *Siebel Retail Finance Installation Guide* for more information

3.1.1 Extracting Resources for Building Branch Teller

- Download the SRFBankingApplicationCommonSoftwareResources.zip file from SRF-BranchTellerExtPackWeblogicASORCLv811.
- Extract SRFBankingApplicationCommonSoftwareResources.zip to d:\ path.

3.2 Banking Application Build Resources

The following topics describe the Siebel Retail Finance banking application build resources:

- Section 3.3, "ANT Installation"
- Section 3.4, "Build Directory Structure"
- Section 3.5, "Class JAR files in the Build Directory"
- Section 3.7, "Branch Teller Statechart Files"

3.3 ANT Installation

Following are the steps in the installation of ANT.

1. Download the ANT 1.8 version from Apache website.
2. Unzip the downloaded ANT 1.8 file into the path - "d:\java". Make sure that ANT folder is extracted under "d:\java" with the name, 'apache-ant-1.8.0'.
3. Set the ANT_HOME environment variable to the above path. That is, d:\java\apache-ant-1.8.0.

3.4 Build Directory Structure

Following table outlines the build directory that is created when you extract the files contained in the Common Software Resources and Build Packs.

Standard Directory Structure

Build Directory	Description
\3rdParty	Contains third-party libraries required for supporting the Banking Application. You must install additional third-party JAR files that are not included with this distribution, see *Siebel Retail Finance Banking Application Installation Guide* for further details.
\Branch	Contains all server-side implementation layer source code, deployment descriptors, resources for the Banking Application, and ANT scripts to compile the War file.
\branch-common	Contains common branch code.
\BranchAdministratorClient	Contains all the front-end code to run the Branch Teller administration application, and an ANT script to compile the Web Archive (WAR) file. **NOTE**: Branch Teller Administrator front-end code is only shipped if the Branch Teller product has been licensed.

Build Directory	Description
\Build	Contains the ANT scripts and other required resources for assembling and compiling the Banking Application and for generating a deployable EAR file.
\Common	Contains the common JAR files and resource files that are shared between Banking Application products.
\Entitlements	Entitlement is a project jar file that contains the source code to take care of privileges, roles, assigning privileges to roles, and so on. This build directory contains the Entitlements source code, deployment descriptors, and ANT scripts to build Entitlements.
\database	Contains the database dump file, and reset scripts for the database.
\Offline	Contains the Offline source code, ANT scripts to compile and build the Branch Offline Server, and embedded version of Oracle Lite database.
\BranchClient	Contains all the front-end code to run the Branch Teller client, and an ANT script to compile the WAR file. **NOTE**: Branch Teller front-end code is only shipped if the Branch Teller product has been licensed.

3.5 Class JAR files in the Build Directory

The Siebel Retail Finance Common Software Resources pack contains several important class JAR files contained in the \siebel\Common\lib folder. The JAR files in this folder are described in the following topics.

3.5.1 Module Layer JAR Files

These JAR files contain compiled classes only (no source code). For ease of future support, you must add these JAR files to the classpath of the extended product when it is being deployed.

Module Layer JAR Files

Module Layer Classes JAR File	Description
branch.jar	Contains all server-side classes, including EJBs, Parameter Objects, Constants and utility classes, for the Module Layer of the Banking Application codebase.
entitlements.jar	Contains module level classes from the Entitlements sub module.
branch-common-fe.jar	Contains all common front end classes used between the Branch Teller client and the Branch Administrator Client.
branch-common.jar	Contains all common classes used between the Branch Teller client and the Branch Server side.
offline.jar	Contains module level classes from the Offline sub-module.

3.5.2 Domain Layer JAR Files

The following table lists the Domain Layer JAR files. There are only compiled classes in these JAR files (no source code). Because these are domain-level JAR files, the classes in these JAR files are altered when extensions are made.

Domain Layer JAR Files

Domain Layer JAR file	Description
branch-impl.jar	Contains the Domain Layer classes for all server-side Banking Application classes: Entity, Session, Parameter/Value/Factory objects, and utility classes. Some or all of the classes in this JAR file might be required for deployment of the generic or extended Banking Application.
entitlements-impl.jar	Contains domain-level classes from the Entitlements submodule.
branch-common-impl.jar	Contains all common Domain Layer classes, shared between the Branch Teller client and the Branch Server side.

3.5.3 Support JAR Files

The following table lists the support JAR files. These JAR files contain compiled classes only (no source code). For ease of future support, you must add these JAR files to the classpath of the extended Banking Application when it is being extended and deployed.

Support JAR Files

Support JAR File	Description
mca.jar	Contains MCA Services classes.
core-bos.jar	Contains the Core and Sector Layer classes for all Banking Application entity beans.
bfa-utils.jar	Contains utility classes.
statemachine-ext.jar	Contains extension classes for the StateSoft Statemachine framework supporting the Screen Orchestrator tool in the Financial Transactions WorkBench.
mca-eabmq.jar	Contains Enterprise Access Bean (EAB) and MQSeries connector helper classes.
offline-impl.jar	Contains the domain layer classes for offline.

3.6 Signing Customized JAR and WAR Files

All customized JAR and WAR files must be signed before the application can be built and deployed. All the JAR and WAR files will be automatically signed as a part of build process itself. If the files have not been signed, a WebStart error will be thrown and deployment will fail. In order to sign the JAR and WAR files a keystore must be generated.

The settings for signing JAR and WAR files are in a siebel\Common\build.properties and must not contain empty strings. The value for the storepass setting must be, at least six characters in length. The following settings in must be configured:

alias=<keystore alias>

storepass=<keystore password>

companyName=<company>

operatingUnit=<operating unit>

organization=<organization>

country=<country>

To generate the keystore:

1. Navigate to the \siebel\Build directory
2. Double-click the JavaPrompt.vbs file
3. Enter the following at the command line:

   cd BranchClient

   build gen-key

   A prompt will appear asking the user to delete the existing keystore, enter Y to proceed.
4. The keystore will be generated to the siebel\Common\KeyStore directory

> **Note:** This command can also be run from the BranchAdministratorClient directory.

3.7 Branch Teller Statechart Files

The Branch Teller client application is implemented as a Swing application using the Screen Orchestrator tool. This tool uses a product statechart that can be extended in client customizations that use the Screen Orchestrator.

The statechart XML files for the Branch Teller components are located in subfolders in the Standard Directory Structure as shown in the following table.

Statechart XML Files and Supporting files for the Banking Application Components

Component	Subfolders
Branch Teller Client	\siebel\BranchClient\statechart
Branch Teller Administrator Client	\siebel\BranchAdministratorClient\statechart

3.8 Downloading and Configuring Log4J

Console Logging is configured and enabled in default configuration. If you want to change logging to Log4j, proceed with the following steps.

1. Download log4j.jar,version 1.2.6, from Apache website http://logging.apache.org/log4j/1.2/.
2. Move the file to siebel\3rdParty\lib\log4j directory.
3. Edit index.jnlp file in Branchclient\webstart to include log4j.jar in webstart file download under the resources tag, as <jar href="log4j.jar" download="eager"/>.
4. Perform the step 3 for BranchAdministratorClient.

5. Edit BranchAdministratorClient\build.xml and remove the comment the below line under pre-sign target,

   <!-- fileset dir="${3rdParty.lib}/log4j" includes="*.jar"/ -->.

6. Perform the step for BranchClient as well.
7. Edit common\resources\eloggerfactory.properties.

   Change the value of

   com.eontec.mca.elogger.factory property

   to

   com.bankframe.services.logger.log4j.LOG4JLoggerFactory

See, Chapter 5, "Oracle WebLogic Build Process" and Chapter 6, "WebSphere Build Process."

3.9 The Java Doc Design Documentation

There is a folder named src in the project, Design. This folder contains the java files that represent a skeleton of the exact structure, name, and all the methods of the Sessions, Business Objects, Value Objects, and Parameter Objects in the Current Base code.

The source files in the folder src under the Design project has to be manually updated to make the design and base code to be in sync.

In order to update the Source files to keep the JAVADOC design docs and base code to be in sync,the following steps must be performed:

1. Open the corresponding file (Sessions, Business Objects, Value Objects, and Parameter Objects) in the src folder under design.
2. Update the file with necessary changes. (Changes could be class description changes, method description changes, or adding new method or attribute to the class.)
3. Save the updated file.
4. Open build prompt (\Build\JavaPrompt.vbs).
5. Change dir to Design (>cd Design).
6. Type ant and press Enter.

This should generate the Java doc html files in the scratch folder and a zipped folder named design_javadoc.zip in the build_release folder.

3.10 How to update the BankframeConstants file

TellerConsntantsKeys.java should not be modified manually. This file is auto-generated from BankframeConstants.properties file. To update the BankframeConstants.properties file, follow these steps.

1. Perform required changes in BankframeConstants.propertes.
2. Check out com.bankframe.util.retail.solutionset.utilities.TellerConstantsKeys.java (\branch-common\Source\Utilities\com\bankframe\util\retail\solutionset\utilities\TellerConstantsKeys.java).
3. Open build prompt (\Build\JavaPrompt.vbs).

4. Change dir to Common (cd Common).
5. Run batch to generate TellerConstantKey.java (build-constants-keys-class.bat).
6. Refresh TellerConstantKeys.java in Eclipse and check it in.

Note: The above procedure is required in the following cases:

- adding a new constant into BankframeConstants.propertes
- modifying name of existing constant
- adding/modifying value(s) of a constant, which is a list (for example, DIRECT_DEBIT_STATUS=String[FIRST_ PAYMENT,ACTIVE,FINAL_ PAYMENT,CLOSED,SUSPENDED,CANCELLED])

If you want to change the value of the existing single-value constant (for example, FEE_PAYMENTS_DESTINATION_ACCOUNT_ NUMBER=String[3399999998]), re-generation of TellerConstantKeys is not required

---

Single-value Constants

Code for single-value constants remains unchanged. For example, TellerConstantsKeysImpl.UNIQUE_ID_LENGTH.

Lists constants

Code for accessing individual values in a list has been changed

The value

- ListOfValuesImpl.getStringValueInList(67, TellerConstantsKeysImpl.JOURNAL_ ENTRY_TYPE);

is changed to

- TellerConstantsKeysImpl.JOURNAL_ENTRY_TYPE_$_WAIVED_FEE

If you need to add a constant into TellerConstantsKeys without corresponding value in BankframeConstants.properties (for example, a constant, which refers to BankframeMessages.properties and is used in validations), you can add such constant to com.bankframe.util.retail.solutionset.utilities.TellerConstants interface. This interface is implemented by auto-generated TellerConstantsKeys

Therefore, the code that uses it remains the same.

3.11 Encrypting and decrypting

This section deals with the following two categories of encryption.

- Encrypting Passwords of Existing Users
- Encrypting User Credentials for BI Publisher

3.11.1 Encrypting Passwords of Existing Users

To encrypt the password for the already existing user, perform the following steps:

1. Open BankframeResource.properties and change the property messageDigest.algorithm=MD2.

2. Run the class MessageDigestUtils in package com.bankframe.services.security, with password as the argument.

   This will generate the encrypted password in the console.

   example: Hash: 10013357114205120322624441233408916413734

3. Update the ACTOR table with this encrypted value.

3.11.2 Encrypting User Credentials for BI Publisher

Siebel Branch Teller 8.1.1 has encrypted the user names and passwords and stored these encrypted values in the BankframeResource.properties file. These details will then be subsequently used to connect BI Publisher from Siebel Teller application.

If you want to change the user credentials for BI Publisher connectivity from Siebel Branch Teller, then have to generate encrypted values for new user name and password. Then, you need to set the encrypted values in the properties file, BankframeResource.properties.

Following properties needs to be changed with new encrypted value.

1. For Teller login
   - oracle.bi.publisher.teller.username
   - oracle.bi.publisher.teller.password
2. For Supervisor login
   - oracle.bi.publisher.supervisor.username
   - oracle.bi.publisher.supervisor.password

Under Build project, encrypt.bat file is used to generate encrypted value. Following are the steps to generate encrypted value:

1. Set keystore file location for property encryption.keystore.file.path in BankframeResource.properties.

   There are different keystores for Oracle Weblogic and IBM WebSphere application server.

   This is due to the usage of different JVM for IBM WebSphere application server when compared to Oracle Weblogic application server.

   - For WebSphere application server, encryption.keystore.file.path = ../Common/resources/IBM/
   - For Weblogic application server, encryption.keystore.file.path = ../Common/resources/

2. To generate encrypted value, go to Build folder and open javaPrompt.vbs file.

   Enter encrypt.bat file and you will be prompted to input the text for which the encrypted value has to be generated.

3.12 Printing Framework

The MCA Services report framework supports high quality form report generation via the Oracle Business Intelligence Publisher (BI Publisher) product. The framework includes a Session EJB which accepts a Vector of DataPackets and generates an report and stores the report content in Report_Data database table. The Report_Data table will return a report id. It is processed by Teller to produce an output to the Internet browser.

The MCA Services printing framework is implemented as a standard two layer Session Bean EJB as follows:

- The solution set provided in the com.bankframe.bp.retail.solutionset.report package.
- The implementation provided in the com.bankframe.bp.retail.solutionset.impl.report package.

3.12.1 com.bankframe.bp.retail.solutionset.report

This package defines the MCA report framework to generate the report content and stores it in Report_Data database table. It returns the report id necessary for teller to view the report from any standard Internet browser. The report id is passed to the Business process which will call the BI Publisher's web service to generate PDF with layout defined in RTF format. The generated PDF report document will be stored in the database as a BLOB field and will enable the Print button. When a user clicks the print button, Siebel Branch Teller will open the Internet browser for viewing the generated PDF report.

It contains the following classes:

GenericReportBean

This class provides all the methods for accepting a vector of DataPackets and store the BI Publisher generated report content in database and send back report_id in response datapacket.

generateReport()

This method has the following signature:

public Vector generateReport(DataPacket dataPacket) throws ProcessingErrorException;

This method accepts a Vector of DataPackets containing the data to be generated from BI Publisher in pdf format. There is a servlet class for viewing report. The servlet class is a part of MCA that will access the teller database using CMP entity bean and query for the generated reports by passing unique report id column value. The PDF report will be stored in BLOB column on the database and the data will be stored in byte array. Therefore, the byte array will be used as data type in entity beans mappings for inserting or fetching blob column value.

3.12.2 Generating the Service

The REQUEST_ID of the first DataPacket in the Vector must have a REQUEST_ID of MC065. For example, from a JSP front end, the following would be specified within the JSP:

```
<FORM NAME="printPage" ACTION="" METHOD="post">
<p>Title
  <input type="text" name="Title">
</p>
<p>First Name
  <input type="text" name="FirstName">
</p>
<p>Surname
  <input type="text" name="Surname">
</p>
```

```
<input type="hidden" name="REQUEST_ID" value="MC065">
<input type="hidden" name="JF_JOB_CARD" value="jobname printername">
<input type="submit" name="Submit" value="Submit">
```

3.12.3 Calling the Service from another Session

When calling the service from another session, a Vector of DataPackets must be passed to the GenericReportBean and the generateReport() method. For example:

```
public Vector generateReport(DataPacket dataPacket) throws
ProcessingErrorException{

Vector response = new Vector();

log.info("Generating Report.");

String reportCatagoryId = (String)dataPacket.get("REPORT_CATEGORY_ID");

String companyCode = (String)dataPacket.get("COMPANY_CODE");

String userId = (String)dataPacket.get("USER_ID");

ReportCategory reportCategory = getReportCategory(reportCatagoryId);

checkReportAccess(reportCategory, companyCode, userId);

String oracleBISessionID = loginToOBIPublisher(reportCategory);

DataPacket reportData =
generateReportFromOBIPublisher(oracleBISessionID,dataPacket, reportCategory);

ReportPOImpl reportPOImpl = new ReportPOImpl(reportData);

saveReport(reportPOImpl);

logoutFromOBI(oracleBISessionID);

DataPacket reportResponseData = new DataPacket("REPORT_INFO");

String reportId = (String)reportData.get("REPORT_ID");

reportResponseData.put("REPORT_ID",reportId);

try {

String reportUrl = "REPORT_ID=" + reportId;

reportUrl = reportUrl + "&sessionId="+ dataPacket.get("sessionId");

reportUrl = reportUrl + "&ENTITLEMENTS_ACTOR_ID="+
dataPacket.get("ENTITLEMENTS_ACTOR_ID");

reportUrl = reportUrl+ "&ENTITLEMENTS_ACCESS_PROVIDER_ID="+
dataPacket.get("ENTITLEMENTS_ACCESS_PROVIDER_ID");

reportUrl = reportUrl + "&ENTITLEMENTS_CHANNEL_ID="+
dataPacket.get("ENTITLEMENTS_CHANNEL_ID");

reportUrl = reportUrl + "&COMPANY_CODE="+ dataPacket.get("COMPANY_
CODE");

String encryptedReportUrl = EncryptMessage.getInstance().encrypt(reportUrl);

reportResponseData.put("ENCRYPTED_REPORT_URL",encryptedReportUrl);

} catch (Exception e1) {
```

throw new ProcessingErrorException(new BankFrameMessage(TellerErrorNumberImpl.REPORT_URL_ERROR,new String[]{" For report id : "+reportId}));

}

response.add(reportResponseData);

log.info("Report Generated.");

return response;

}

3.12.4 Oracle BI Publisher Architecture

- Creation of RTF Template
- OBI Publisher Configuration
- Set up Webservice for Report generation
- Creation of XSL style sheets
- Database Configuration
- Calling BI Publisher exposed web service from Siebel Teller application
- Viewing generated report through Internet browser by accessing web application
- Localization

BankframeResource.properties holds properties relating to the BI Publisher connectivity, generation and retrieving of report. These properties are:

oracle.bi.publisher.ws.endpoint.url=http://localhost:7001/xmlpserver/services/PublicReportService

teller.report.url = http://localhost:7001/BankFrameMCA/ReportServlet

encryption.keystore.file.path=../Common/resources/IBM

oracle.bi.publisher.teller.username={DES}599108D7AA68BDC3

oracle.bi.publisher.teller.password={DES}599108D7AA68BDC3

Detailed Process

For configuring and implementing the BI publisher for CTR printing, perform the following steps:

1. Configure report in BI Publisher. Upload the report template and create layout. For developer testing, sample data file can be uploaded.
2. Utilize webservice and its operations exposed by BI Publisher(*http://hostname:port/xmlpserver/services/PublicReportService)*
3. Steps to follow to generate reports in BI Publisher using exposed webservice are as follows.
   - Login to BI Publisher through web service login operation. User credential and end URL location will be stored in bankframeresource properties file.
   - After successful login, fetch parameters from database like format, layout report file, access, and so on. Thus if changes are required in any format or layout, it will not require any change in source code.

- Generated report response from webservice will be stored in teller database as blob column and content of this column will be a byte array value.
- Teller will have the option to invoke web application through URL which will be formed with unique field report id value.
- To invoke web application, teller must have valid session id and Web application will fetch report content from the database and present in the Internet browser.

3.13 Bringing the application to offline without shutting down the central server

To bring the application to offline without shutting down the central server, the OfflineTest file is modified.

The default location is C:\OfflineTest incase of windows and ./OfflineTest incase of Linux.

This modification enables a switch to offline mode even when the application server is running. The location of the OfflineTest file is specified by the value of the-Doffline.helper.test.file setting in the OfflineServerController.bat file or the OfflineServerController.sh file.

Modifying the online/offline setting enables testing to switch between online and offline mode without stopping the application server. The OfflineTest file has two entries. You can switch between online and offline modes by swapping the comment character (#) at the start of the applicable line.

For example, the following configuration results in an offline response:

com.siebel.rf.offline.ei.channel.client.AlwaysOfflineClient

#com.siebel.rf.webstart.client.WebStartChannelClient

Note: When the application is run from a client machine there are two OfflineTest file. One in the location where the branch server is running and the other in the client machine. Change both the files to bring the application to offline.

4 Database Updates

This chapter covers the details regarding the updates to the databases and includes the following topics.

- Section 4.1, "Creation of Database, Tablespace, and Bufferpool"
- Section 4.2, "Database Changes in Release 8.1.1"

4.1 Creation of Database, Tablespace, and Bufferpool

The following commands are specific to DB2.

The command to create database is:

- db2 create database SBT_QA4 pagesize 16 K

The command to create tablespace is:

- db2 create bufferpool SBT_BF_POOL size 16 pagesize 16 K

The command to create bufferpool is:

- db2 create tablespace SBT_SPACE pagesize 16 K bufferpool SBT_BF_POOL

Note: In these commands, sample dbname, tablespace name, and bufferpool name are used. You can use your own names.

4.2 Database Changes in Release 8.1.1

The new tables and attributes in release 8.1.1 are listed here. These changes apply to both Oracle and DB2.

4.2.1 BI Publisher

The new tables in BI Publisher, related to reports are:

- REPORT_DATA
- REPORT_CATEGORY
- SYSTEMPROPERTIES

4.2.2 General Ledger

In the INTERNAL_ACCOUNT table, two new attributes named, ACCOUNT_NAME and CURRENCY are added. Therefore, together with account numbers, account names and currency are also present now. An existing attribute, LEDGER_TYPE, present in this table has been modified.

4.2.3 International Transfer and Wire transfer

The new tables in International Transfer and Wire Transfer, related to details regarding IBAN and ABA are:

- IBAN_STRUCTURE
- ABA_DETAILS

4.2.4 Supervisor Override

The new table in Supervisor Override, related to supervisor referral reason is:

- SUPERVISOR_REF_REASON

In the BRANCH table, a new attribute named, REGION_CODE is added. Therefore, the supervisor override request goes to all regions.

In the PRIVILEGE table, a new attribute named, OVERRIDE_LIMIT_ID is added. Therefore, now, the override limit can be specified.

In the SUPERVISOR_REF_OVERRIDE table, new attributes named, FEE_AMOUNT, FEE_CURRENCY, and FEE_WAIVED_INDICATOR are added. Therefore, now, the fee amount, fee currency, and waiver if required, can be specified.

4.2.5 Incomplete Customer Session

The new table in Incomplete Customer Session, for storing incomplete customer session details is:

- CUSTOMER_SESSION

  IS_TRANSFERRED and TARGET_TELLER_ID attributes in this table differentiates a target teller from general tellers, with transferred session.

4.2.6 Cashless User with Cash Transactions

In the TELLER table, a new attribute named, CASH_TRANS_ALLOWED is added which identifies a teller who can perform cash transaction.

4.2.7 Modified Attributes

The attributes that are not newly introduced, but modified in the tables are listed here.

- DEST_ACC_NO - The column length of this attribute has been increased in following three tables - JOURNL_ITEM, RETAIL_MOVEMENTS, and TRANS_ DETAILS.
- PIN_NO - The column length of this attribute has been increased the table, BANK_CARD.
- REQUEST AND RESPONSE - The datatypes of these attributes has been modified to allow extra information to be stored, in the table, AUDIT_TRAIL.

5

Oracle WebLogic Build Process

This chapter covers building Siebel Branch Teller Modules for the Oracle WebLogic Application Server, and includes the following topics:

- Section 5.1, "Oracle WebLogic Build Process Prerequisite"
- Section 5.2, "SBT Build Resources for Deployment on the Oracle WebLogic"

5.1 Oracle WebLogic Build Process Prerequisite

When building an EAR file for deployment on the Oracle WebLogic Application Server the Oracle WebLogic Application Server must be installed on the build machine. It is assumed that the installation is done at d:\bea.

5.1.1 Verifying the Build environment

We need to ensure that the settings are setup up correctly in:

siebel\Build\build.cmd

The Weblogic target calls d:\bea\wlserver_10.3\server\bin\setWLSenv.cmd

siebel\Build\java\javaenv.bat

BEA_WL_HOME=d:\bea

ANT_HOME=d:\java\apache-ant-1.8.0

JAVA_HOME=d:\java\jdk1.6.0_18

5.2 SBT Build Resources for Deployment on the Oracle WebLogic

The following build resources are provided in the \siebel\Build folder:

- A Windows script, JavaPrompt.vbs, which creates a command prompt that has the CLASSPATH and PATH variables configured to invoke the Banking Application build process. It also assumes that JDK is installed at d:\java\jdk1.6.x. The script, JavaPrompt.vbs includes the variable configuration of JAVA_HOME, ANT_HOME, ANT_LIB, and BEA_HOME.
- A build script, build-all.xml, which builds all the projects in the correct order and creates the EAR file in the d:\temp\build_tmp\build_release\ folder.
- The file build-all.xml builds the projects in the following order:

  a. Branch

  b. Offline BranchAdministratorClient

c. BranchClient

d. Build

Note: The precompiled Entitlements project is located in the Siebel\Build\resource folder and the build-all script does not rebuild this project. Refer to the topic Building Individual Siebel Branch Teller Projects for Deployment on WebSphere to rebuild the Entitlements project.

5.2.1 Building SBT Projects and EAR for Deployment

This topic covers rebuilding all the Siebel Branch Teller banking application projects, and rebuilding the EAR file, for deployment on the Oracle WebLogic Application Server.

To build the Siebel Retail Finance EAR for deployment on Oracle WebLogic

1. Navigate to the \siebel\Build directory.
2. Double-click the JavaPrompt.vbs file.
3. Enter the following at the command prompt:

   cd Build

   buildall weblogic

5.2.2 Building Entitlements Project for Deployment

This topic covers building the Entitlements project for deployment on the Oracle WebLogic Application Server.

To build the Entitlements project for deployment on the Oracle WebLogic Application Server

1. Navigate to the \siebel\Build directory.
2. Double-click the JavaPrompt.vbs file.
3. Enter the following at the command prompt:

   cd Entitlements

   build weblogic

5.2.3 Building Branch Project for Deployment

This topic covers building the Branch project for deployment on the Oracle WebLogic Application Server.

To build the Branch project for deployment on the Oracle WebLogic Application Server

1. Navigate to the \siebel\Build directory.
2. Double-click the JavaPrompt.vbs file.
3. Enter the following at the command prompt:

   cd Branch

   build weblogic

5.2.4 Building Offline Branch Project for Deployment

This topic covers building the Offline Branch project for deployment on the Oracle WebLogic Application Server.

To build the Offline Branch project for deployment on the Oracle WebLogic Application Server

1. Navigate to the \siebel\Build directory.
2. Double-click the JavaPrompt.vbs file.
3. Enter the following at the command prompt:

   cd Branch

   build weblogic

5.2.5 Building BranchAdministratorClient Project for Deployment

This topic covers building the BranchAdministratorClient project for deployment on the Oracle WebLogic Application Server.

To build the BranchAdministratorClient project for deployment on the Oracle WebLogic Application Server

1. Navigate to the \siebel\Build directory.
2. Double-click the JavaPrompt.vbs file.
3. Enter the following at the command prompt:

   cd BranchAdministratorClient

   build weblogic

The ANT build process creates branchadmin.war. The wars and ears are generated into d:\temp\build_tmp\build_staging.

5.2.6 Building BranchClient Project for Deployment

This topic covers building the BranchClient project for deployment on the Oracle WebLogic Application Server.

To build the BranchClient project for deployment on the Oracle WebLogic Application Server

1. Navigate to the \siebel\Build directory.
2. Double-click the JavaPrompt.vbs file.
3. Enter the following at the command prompt:

   cd BranchClient

   build weblogic

The ANT build process creates the branchteller.war file. The wars and ears are generated into d:\temp\build_tmp\build_staging.

5.2.7 Building SBT EAR for Deployment

This topic covers rebuilding the EAR file for deployment on the Oracle WebLogic Application Server.

To build the EAR file for deployment on the Oracle WebLogic Application Server

1. Navigate to the \siebel\Build directory.
2. Double-click the JavaPrompt.vbs file.
3. Enter the following at the command prompt:

   cd Build

   build weblogic

6 WebSphere Build Process

This chapter covers building Siebel Branch Teller Modules for the WebSphere application server, and includes the following topics:

- Section 6.1, "WebSphere Build Process Prerequisites"
- Section 6.2, "SBT Build Resources for Deployment on the WebSphere"

6.1 WebSphere Build Process Prerequisites

The following prerequisites apply when building an EAR file for deployment on the WebSphere application server:

- The WebSphere application server must be installed on the build machine.
- "The \siebel\Common\build.properties file must be configured for websphere.home =<WebSphere installation root>. The build scripts assume that the WebSphere installation root is d:\WAS\AppServer."
- The \siebel\deploy\build.properties file must be configured as follows for a DB2 database:

  a. db.driver.name= com.ibm.db2.jcc.DB2Driver

  b. db.url.prefix= jdbc:db2://

  c. target.dbtype.oracle=db2

6.2 SBT Build Resources for Deployment on the WebSphere

The following build resources are provided in the \siebel\Build folder:

- A Windows script, JavaPrompt.vbs, which creates a command prompt that has the CLASSPATH and PATH variables configured to invoke the Banking Application build process. It also assumes that JDK is installed at d:\java\jdk1.6.0_18.
- A build script, build-all.xml, which builds all the projects in the correct order and creates the EAR file in the d:\temp\build_tmp\build_release\ folder.
- The file build-all.xml builds the projects in the following order:

  a. Branch

  b. Offline BranchAdministratorClient

  c. BranchClient

  d. Build

Note: The precompiled Entitlements project is located in the Siebel\Build\resource folder and the build-all script does not rebuild this project. Refer to the topic Building Individual Siebel Branch Teller Projects for Deployment on WebSphere to rebuild the Entitlements project.

6.2.1 Building SBT Projects and EAR for Deployment

This topic covers rebuilding all the Siebel Branch Teller banking application projects, and rebuilding the EAR file, for deployment on the WebSphere Application Server.

1. Navigate to the \siebel\Build directory.
2. Double-click the JavaPrompt.vbs file.
3. If you have a DB2 database, enter the following at the command prompt:

   cd Build

   buildall websphere-db2

6.2.2 Building Entitlements Project for Deployment

This topic covers building the Entitlements project for deployment on the WebSphere Application Server.

To build the Entitlements project for deployment on WebSphere

1. Navigate to the \siebel\Build directory.
2. Double-click the JavaPrompt.vbs file.
3. If you have a DB2 database, enter the following at the command prompt:

   cd Entitlements

   build websphere-db2

6.2.3 Building Branch Project for Deployment

This topic covers building the Branch project for deployment on the WebSphere Application Server.

To build the Branch project for deployment on WebSphere

1. Navigate to the \siebel\Build directory.
2. Double-click the JavaPrompt.vbs file.
3. If you have a DB2 database, enter the following at the command prompt:

   cd Branch

   build websphere-db2

6.2.4 Building Offline Branch Project for Deployment

This topic covers building the Offline Branch project for deployment on the WebSphere Application Server.

To build the Offline Branch project for deployment on WebSphere

1. Navigate to the \siebel\Build directory.

2. Double-click the JavaPrompt.vbs file.
3. If you have a DB2 database, enter the following at the command prompt:

   cd Branch

   build websphere-db2

6.2.5 Building BranchAdministratorClient Project for Deployment

This topic covers building the BranchAdministratorClient project for deployment on the WebSphere Application Server.

To build the BranchAdministratorClient project for deployment on WebSphere

1. Navigate to the \siebel\Build directory.
2. Double-click the JavaPrompt.vbs file.
3. Enter the following at the command prompt:

   cd BranchAdministratorClient

   build

The ANT build process creates the branchadmin.war file. The ears and wars are generated to d:\temp\build_tmp\build_staging.

6.2.6 Building BranchClient Project for Deployment

This topic covers building the BranchClient project for deployment on the WebSphere Application Server.

To build the BranchClient project for deployment on WebSphere

1. Navigate to the \siebel\Build directory.
2. Double-click the JavaPrompt.vbs file.
3. Enter the following at the command prompt:

   cd BranchClient

   build

The ANT build process creates the branchteller.war file. The ears and wars are generated to d:\temp\build_tmp\build_staging.

6.2.7 Building SBT EAR for Deployment

This topic covers rebuilding the EAR file for deployment on the WebSphere Application Server.

To build the EAR file for deployment on WebSphere

1. Navigate to the \siebel\Build directory.
2. Double-click the JavaPrompt.vbs file.
3. If you have a DB2 database, enter the following at the command prompt:

   cd Build

   build websphere-db2

This process will build the application to the local machine. If you need to build to another machine, you can follow the instructions in the *Siebel Retail Finance Banking Application Installation Guide*.

Note: A full build can run for a considerable length of time—typically 30-40 minutes on a 2 gigahertz (GHz) or faster PC with 512 megabyte (MB) RAM.