Oracle® iStore

API Reference Guide

Release 11*i*

April 2002

Part No. A95199-02

ORACLE®

Oracle iStore API Reference Guide, Release 11*i*

Part No. A95199-02

Copyright © 2001, 2002, Oracle Corporation. All rights reserved.

The Programs (which include both the software and documentation) contain proprietary information of Oracle Corporation; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. Oracle Corporation does not warrant that this document is error free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Oracle Corporation.

If the Programs are delivered to the U.S. Government or anyone licensing or using the programs on behalf of the U.S. Government, the following notice is applicable:

Restricted Rights Notice Programs delivered subject to the DOD FAR Supplement are "commercial computer software" and use, duplication, and disclosure of the Programs, including documentation, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement. Otherwise, Programs delivered subject to the Federal Acquisition Regulations are "restricted computer software" and use, duplication, and disclosure of the Programs shall be subject to the restrictions in FAR 52.227-19, Commercial Computer Software - Restricted Rights (June, 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and Oracle Corporation disclaims liability for any damages caused by such use of the Programs.

Oracle is a registered trademark of Oracle Corporation. Other names may be trademarks of their respective owners.

Contents

Send Us Your Comments

Oracle iStore API Reference Guide, Release 11*i*

Part No. A95199-02

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: eccontent_us@oracle.com
- FAX: (650) 654-6208 Attn: Oracle iStore Documentation
- Postal service:
  Oracle Corporation
  Oracle iStore Documentation
  500 Oracle Parkway, 6op4
  Redwood Shores, CA 94065
  USA

If you would like a reply, please give your name, address, telephone number, and (optionally) electronic mail address.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

Audience for This Guide

Welcome to Release 11*i* of *Oracle iStore API Reference Guide*.

This guide assumes you have a working knowledge of the following:

- The principles and customary practices of your business area.
- Oracle iStore 11*i*

  If you have never used Oracle iStore 11*i*, Oracle suggests you attend one or more of the Oracle iStore 11*i* training classes available through Oracle University.
- The Oracle Applications graphical user interface.

  To learn more about the Oracle Applications graphical user interface, read the *Oracle Applications User's Guide*.

See Other Information Sources for more information about Oracle Applications product information.

How To Use This Guide

This guide contains the information you need to understand and use Oracle iStore 11*i*.

- Chapter 1 lists the supported Oracle iStore 11*i* Store Java APIs.
- Chapter 2 lists the supported Oracle iStore 11*i* Display Manager Java APIs.
- Chapter 3 lists the supported Oracle iStore 11*i* Catalog Java APIs.
- Chapter 4 lists the supported Oracle iStore 11*i* Shopping Cart Quote APIs.

- Chapter 5 lists the supported Oracle iStore 11*i* Postsales APIs.
- Appendix A details standards for customizing Oracle iStore 11*i* JavaServer Pages™ (JSP™).
- Appendix B explains how to set up Oracle JDeveloper as an Oracle iStore 11*i* custom development environment.

Typographic Conventions

This manual uses the typographic conventions listed in the following table:

Convention	Meaning
italic text	Book titles
`Courier text`	User commands, file content examples, directory names
UPPERCASE	Structured Query Language (SQL) commands, initialization parameters, profile options, responsibilities, or environment variables
boldface text	Menu, button, keyboard, and form options, emphasis
< >	Angle brackets enclose user-supplied names. Note: Do not type the angle brackets.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. 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. Standards will continue to evolve over time, and Oracle Corporation 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 additional information, visit the Oracle Accessibility Program Web site at http://www.oracle.com/accessibility/.

Accessibility of Code Examples in Documentation

JAWS, a Windows screen reader, 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, JAWS may not always read a line of text that consists solely of a bracket or brace.

Other Information Sources

You can choose from many sources of information, including online documentation, training, and support services, to increase your knowledge and understanding of Oracle iStore 11*i*.

If this guide refers you to other Oracle Applications documentation, use only the Release 11*i* versions of those guides.

Online Documentation

All Oracle Applications documentation is available online (HTML or PDF). Online help patches are available on MetaLink.

Related Documentation

Oracle iStore 11*i* shares business and setup information with other Oracle Applications products. Therefore, you may want to refer to other product documentation when you set up and use Oracle iStore 11*i*.

You can read the documents online by choosing Library from the expandable menu on your HTML help window, by reading from the Oracle Applications Document Library CD included in your media pack, or by using a Web browser with a URL that your system administrator provides.

If you require printed guides, you can purchase them from the Oracle Store at http://oraclestore.oracle.com.

Documents Related to All Products

Oracle Applications User's Guide

This guide explains how to enter data, query, run reports, and navigate using the graphical user interface (GUI) available with this release of Oracle iStore 11*i* (and any other Oracle Applications products). This guide also includes information on setting user profiles, as well as running and reviewing reports and concurrent processes.

You can access this user's guide online by choosing "Getting Started with Oracle Applications" from any Oracle Applications help file.

Documents Related to This Product

Oracle iStore Implementation Guide

This guide has the information needed to implement Oracle iStore 11*i*, including procedures for setting up and customizing Oracle iStore 11*i*, and profile option descriptions.

Oracle iStore User Guide

This document provides users with information on general principles and procedures for maintaining Web stores using Oracle iStore 11*i*.

Oracle eTechnical Reference Manuals

Each eTechnical Reference Manual (eTRM) contains database diagrams and a detailed description of database tables, forms, reports, and programs for a specific Oracle Applications product. This information helps you convert data from your existing applications, integrate Oracle Applications data with non-Oracle applications, and write custom reports for Oracle Applications products. Oracle eTRM is available on Metalink.

Training and Support

Training

Oracle offers training courses to help you and your staff master Oracle iStore 11*i* and reach full productivity quickly. You have a choice of educational environments. You can attend courses offered by Oracle University at any one of our many Education Centers, you can arrange for our trainers to teach at your facility, or you can use Oracle Learning Network (OLN), Oracle University's online education utility. In addition, Oracle training professionals can tailor standard courses or develop custom courses to meet your needs. For example, you may want to use your organization's structure, terminology, and data as examples in a customized training session delivered at your own facility.

Support

From on-site support to central support, our team of experienced professionals provides the help and information you need to keep Oracle iStore 11*i* working for you. This team includes your Technical Representative, Account Manager, and Oracle's large staff of consultants and support specialists with expertise in your business area, managing an Oracle8*i* server, and your hardware and software environment.

Oracl*eMetaLink*

Oracl*eMetaLink* is your self-service support connection with web, telephone menu, and e-mail alternatives. Oracle supplies these technologies for your convenience, available 24 hours a day, 7 days a week. With Oracl*eMetaLin*k, you can obtain information and advice from technical libraries and forums, download patches, download the latest documentation, look at bug details, and create or update TARs. To use MetaLink, register at (http://metalink.oracle.com).

Alerts: You should check Oracl*eMetaLink* alerts before you begin to install or upgrade any of your Oracle Applications. Navigate to the Alerts page as follows: Technical Libraries/ERP Applications/Applications Installation and Upgrade/Alerts.

Self-Service Toolkit: You may also find information by navigating to the Self-Service Toolkit page as follows: Technical Libraries/ERP Applications/Applications Installation and Upgrade.

Do Not Use Database Tools to Modify Oracle Applications Data

Oracle STRONGLY RECOMMENDS that you never use SQL*Plus, Oracle Data Browser, database triggers, or any other tool to modify Oracle Applications data unless otherwise instructed.

Oracle provides powerful tools you can use to create, store, change, retrieve, and maintain information in an Oracle database. But if you use Oracle tools such as SQL*Plus to modify Oracle Applications data, you risk destroying the integrity of your data and you lose the ability to audit changes to your data.

Because Oracle Applications tables are interrelated, any change you make using Oracle Applications can update many tables at once. But when you modify Oracle Applications data using anything other than Oracle Applications, you may change a row in one table without making corresponding changes in related tables. If your tables get out of synchronization with each other, you risk retrieving erroneous information and you risk unpredictable results throughout Oracle Applications.

When you use Oracle Applications to modify your data, Oracle Applications automatically checks that your changes are valid. Oracle Applications also keeps track of who changes information. If you enter information into database tables using database tools, you may store invalid information. You also lose the ability to track who has changed your information because SQL*Plus and other database tools do not keep a record of changes.

About Oracle

Oracle Corporation develops and markets an integrated line of software products for database management, applications development, decision support, and office automation, as well as Oracle Applications, an integrated suite of more than 160 software modules for financial management, supply chain management, manufacturing, project systems, human resources and customer relationship management.

Oracle products are available for mainframes, minicomputers, personal computers, network computers and personal digital assistants, allowing organizations to integrate different computers, different operating systems, different networks, and even different database management systems, into a single, unified computing and information resource.

Oracle is the world's leading supplier of software for information management, and the world's second largest software company. Oracle offers its database, tools, and applications products, along with related consulting, education, and support services, in over 145 countries around the world.

1 Oracle iStore 11*i* Store APIs

This chapter contains the following information on the Oracle iStore 11*i* Store public class APIs in the package oracle.apps.ibe.store:

- Store API Class Summary
- Class StoreMinisite
- Class StoreProfile

1.1 Store API Class Summary

APIs for the Oracle iStore 11*i* Store procedures are located in the package oracle.apps.ibe.store. The table below describes the classes briefly.

Table 1–1 Store Class Summary

Class Name	Description
Class StoreMinisite	StoreMinisite provides utility functions to access information about or related to minisites.
Class StoreProfile	StoreProfile contains all the utility functions to get the value of the various profiles in iStore.

1.2 Class StoreMinisite

```
java.lang.Object > oracle.apps.ibe.minisites.Minisite >
oracle.apps.ibe.store.StoreMinisite
```

public class **StoreMinisite**

extends Minisite

StoreMinisite provides utility functions to access information about or related to minisites.

1.2.1 Methods for Class StoreMinisite

The following table is an index of Class StoreMinisite methods:

Table 1–2 Method Index for Class StoreMinisite

Method	Description
getAccessibleMinisiteResponsibilities	Get the list of minisite and responsibility association object for a given user ID, party ID (of type 'ORGANIZATION'), and application ID.
getAccessibleResponsibilities	Get the list of responsibility object for a given minisite ID, user ID, party ID (of type 'ORGANIZATION'), and application ID.
getAccessName	Get the programmatic access name of the minisite.
getBusinessPriceListID	Get the business price list ID for the minisite. Also receives currency code as a parameter.
getCurrencies	Get the list of currencies supported by the minisite.

Table 1–2 Method Index for Class StoreMinisite (Cont.)

Method	**Description**
getDefaultCurrencyCode	Get the default currency code for the minisite.
getDefaultLanguageCode	Get the default language code for the minisite.
getDescription	Get the description of the minisite based on the language of the current session.
getExcludedItems	Get the Excluded Items for the minisite.
getExcludedSections	Get the Excluded Sections for the minisite.
getLanguageCodes	Get the list of language codes supported by the minisite.
getMinisiteID	Get the minisite ID for a given minisite programmatic access name.
getMinisiteResponsibility	Given a minisite ID and responsibility ID, application ID, return the corresponding MinisiteResponsibilityObject.
getMinisiteResponsibilityDisplayName	Given a minisite ID, responsibility and application ID combination, and language code, return the display name for the minisite and responsibility association.
getName	Get the name of the minisite, based on the language of the current session.
getPriceListID	Get the price list ID based on the currency code and user type for the current session.
getRegisteredPriceListID	Get the registered price list ID for the minisite. Also receives currency code as a parameter.
getRootSectionID	Get the root section ID for the minisite.
getWalkinPriceListID	Get the walkin price list ID for the minisite. Also receives currency code as a parameter.
isWalkinAllowed	Check if walkin is allowed in the minisite.

getAccessibleMinisiteResponsibilities

```
public static MinisiteResponsibilityObject[]
getAccessibleMinisiteResponsibilities(BigDecimal pUserID, BigDecimal
pOrgPartyID, BigDecimal pAppID, BigDecimal pCurrRespID, int pLoadRespType,
boolean pCheckPartyAccess)
throws FrameworkException, SQLException
```

Get the list of minisite and responsibility association object for a given user ID, party ID (of type 'ORGANIZATION'), and application ID. Responsibility check and Party access check will be performed depending on the input parameters pLoadRespType and pCheckPartyAccess. If all the minisite responsibilities needs to be loaded irrespective of user's responsibility, then the pLoadRespType should be LOAD_ALL_RESPS. If only those minisite responsibilities needs to be loaded to which the user belongs (i.e. user has the responsibility in every minisite responsibility combination of the list returned), then the type should be LOAD_COMMON_RESPS. If only those minisite responsibilities needs to be loaded where the responsibililty matches with the passed in current responsibility ID, then the type should be LOAD_CURRENT_RESP. Note that the responsibility ID pCurrRespID should belong to application pAppID. If pLoadRespType is LOAD_NORMAL_RESPS, then the list of minisite responsibility combination returned are the ones to which user belongs to (for minisites with resp_access_flag = 'Y') and the ones to which he may not belong to (for minisites with resp_access_flag = 'N'). Party access check will be performed only if pCheckPartyAccess is set to true. Only those objects (combination of minisite and responsibility) will be returned which has responsibility belonging to the application ID specified by pAppID.

Parameters: pUserID - the ID of user

pOrgPartyID - the party ID (of type 'ORGANIZATION) of the user. If the user is of party type 'PERSON' (B2C case), then this value will be null.

pAppID - the application ID

pCurrRespID - the current responsibility ID of the user pUserID. This responsibility should belong to application ID pAppID. This parameter is used only when pLoadRespType is LOAD_CURRENT_RESP.

pLoadRespType - the loading type for the responsibility. If all the minisite responsibilities needs to be loaded irrespective of user's responsibility, then the type should be LOAD_ALL_RESPS. If only those minisite responsibilities needs to be loaded to which user belongs to (i.e. user has the responsibility in every minisite responsibility combination of the list returned), then the type should be LOAD_COMMON_RESPS. If only those minisite responsibilities needs to be loaded where the responsibililty matches with the passed in current responsibility ID, then the type should be LOAD_CURRENT_RESP. If loading type is LOAD_NORMAL_RESPS, then the list of minisite responsibility combination returned are the ones to which user belongs to (for minisites with resp_access_flag = 'Y') and the ones to which he may not belong to (for minisites with resp_access_flag = 'N').

pCheckPartyAccess - to check for party access or not. If false, then no party access check will be performed

Returns: the array of MinisiteResponsibilityObject object which is association between minisite and responsibility. If the user does not have access to any combination of minisite and responsibility, then the API will return an array with length 0.

Throws: FrameworkException if error in getting a connection to the database

Throws: SQLException if error in communicating with the database

getAccessibleResponsibilities

```
public static ResponsibilityObject[] getAccessibleResponsibilities(BigDecimal
pMsiteID, BigDecimal pUserID, BigDecimal pOrgPartyID, BigDecimal pAppID,
BigDecimal pCurrRespID, int pLoadRespType, boolean pCheckPartyAccess)
throws FrameworkException, SQLException
```

Get the list of responsibility object for a given minisite ID, user ID, party ID (of type 'ORGANIZATION'), and application ID. Responsibility check and Party access check will be performed depending on the input parameters pLoadRespType and pCheckPartyAccess. If all the minisite responsibilities needs to be loaded irrespective of user's responsibility, then the pLoadRespType should be LOAD_ALL_RESPS. If only those minisite responsibilities needs to be loaded to which the user belongs (i.e. user has the responsibility in every minisite responsibility combination of the list returned), then the type should be LOAD_COMMON_RESPS. If only those minisite responsibilities needs to be loaded where the responsibililty matches with the passed in current responsibility ID, then the type should be LOAD_CURRENT_RESP. Note that the responsibility ID pCurrRespID should belong to application pAppID. If pLoadRespType is LOAD_NORMAL_RESPS, then the list of minisite responsibility combination returned are the ones to which user belongs to (if minisite has resp_access_flag = 'Y') or the ones to which he may not belong to (if minisite has resp_access_flag = 'N'). Party access check will be performed only if pCheckPartyAccess is set to true. Only those objects (combination of minisite and responsibility) will be returned which has responsibility belonging to the application ID specified by pAppID.

Parameters: pUserID - the ID of user

pOrgPartyID - the party ID (of type 'ORGANIZATION) of the user. If the user is of party type 'PERSON' (B2C case), then this value will be null.

pAppID - the application ID

pCurrRespID - the current responsibility ID of the user pUserID. This responsibility should belong to application ID pAppID. This parameter is used only when pLoadRespType is LOAD_CURRENT_RESP.

pLoadRespType - the loading type for the responsibility. If all the minisite responsibilities needs to be loaded irrespective of user's responsibility, then the type should be LOAD_ALL_RESPS. If only those minisite responsibilities needs to be loaded to which user belongs to (i.e. user has the responsibility in every minisite responsibility combination of the list returned), then the type should be LOAD_COMMON_RESPS. If only those minisite responsibilities needs to be loaded where the responsibililty matches with the passed in current responsibility ID, then the type should be LOAD_CURRENT_RESP. If loading type is LOAD_NORMAL_RESPS, then the list of minisite responsibility combination returned are the ones to which user belongs to (for minisites with resp_access_flag = 'Y') and the ones to which he may not belong to (for minisites with resp_access_flag = 'N').

pCheckPartyAccess - to check for party access or not. If false, then no party access check will be performed

Returns: the array of MinisiteResponsibilityObject object which is association between minisite and responsibility. If the user doesn't has access to any combination of minisite and responsibility, then the API will return an array with length 0.

Throws: FrameworkException if error in getting a connection to the database

Throws: SQLException if error in communicating with the database

getAccessName

```
public static String getAccessName()
throws MinisiteNotExistsException, MinisiteException, SQLException,
FrameworkException
```

Get the programmatic access name of the minisite. This API assumes that iStore cookies are set when calling the API.

Returns: the programmatic access name of the minisite

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteException; occurs in case of application error

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getBusinessPriceListID

```
public static BigDecimal getBusinessPriceListID()
throws MinisiteNotExistsException, MinisiteException, SQLException,
FrameworkException
```

Get the business price list ID for the minisite. The price list ID returned is for the currency in the current session. This API assumes that iStore cookies are set when calling the API.

Returns: business price list ID for the minisite.

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteException; occurs in case of application error

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getBusinessPriceListID

```
public static BigDecimal getBusinessPriceListID(String pCurrencyCode)
throws MinisiteNotExistsException, MinisiteException, SQLException,
FrameworkException
```

Get the business price list ID for the minisite for the given currency code. This API assumes that iStore cookies are set when calling the API.

Parameters: pCurrencyCode - currency code for which the business price list needs to be returned.

Returns: business price list ID for the minisite for the given currency code.

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteException; occurs in case of application error

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getCurrencies

```
public static Hashtable getCurrencies()
throws MinisiteNotExistsException, SQLException, FrameworkException
```

Get the list of currencies supported by the minisite. This API assumes that iStore cookies are set when calling the API.

Returns: Hashtable with the list of currencies.

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getDefaultCurrencyCode

```
public static String getDefaultCurrencyCode()
throws MinisiteNotExistsException, SQLException, FrameworkException
```

Get the default currency code for the minisite. This API assumes that iStore cookies are set when calling the API.

Returns: default currency code for the minisite.

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getDefaultLanguageCode

```
public static String getDefaultLanguageCode()
throws MinisiteNotExistsException, SQLException, FrameworkException
```

Get the default language code for the minisite. This API assumes that iStore cookies are set when calling the API.

Returns: default language code for the minisite.

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getDescription

```
public static String getDescription()
throws MinisiteNotExistsException, MinisiteException, SQLException,
FrameworkException
```

Get the description of the minisite based on the language of the current session. If the description in the language for the current session does not exist, the description in the default language is returned. This API assumes that iStore cookies are set when calling the API.

Returns: the description of the minisite

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteException; occurs in case of application error

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getExcludedItems

```
public static Hashtable getExcludedItems()
throws MinisiteNotExistsException, SQLException, FrameworkException
```

Get the Excluded Items for the minisite. This API assumes that iStore cookies are set when calling the API.

Returns: Hashtable of item IDs

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getExcludedSections

```
public static Hashtable getExcludedSections()
throws MinisiteNotExistsException, SQLException, FrameworkException
```

Get the Excluded Sections for the minisite. This API assumes that iStore cookies are set when calling the API.

Returns: Hashtable of section IDs

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getLanguageCodes

```
public static Hashtable getLanguageCodes()
throws MinisiteNotExistsException, SQLException, FrameworkException
```

Get the list of language codes supported by the minisite. This API assumes that iStore cookies are set when calling the API.

Returns: Hashtable with the list of language codes.

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getMinisiteID

```
public static BigDecimal getMinisiteID(String pAccessName)
throws FrameworkException, SQLException
```

Get the minisite ID for a given minisite programmatic access name. If the access name is null, then the API will return null for minisite ID.

Parameters: pAccessName - programmatic access name of the minisite

Returns: the minisite ID for the corresponding programmatic access name

Throws: FrameworkException if error in getting a connection to the database

Throws: SQLException if error in communicating with the database

getMinisiteResponsibility

```
public static MinisiteResponsibilityObject getMinisiteResponsibility(BigDecimal
pMsiteID, BigDecimal pRespID, BigDecimal pAppID)
throws FrameworkException, SQLException
```

Given a minisite ID and responsibility ID, application ID, return the corresponding MinisiteResponsibilityObject.

Parameters: pMsiteID - the minisite ID

pRespID - the responsibility ID

pAppID - the application ID

Returns: MinisiteResponsibility object

Throws: FrameworkException if error in getting a connection to the database

Throws: SQLException if error in communicating with the database

getMinisiteResponsibilityDisplayName

```
public static String getMinisiteResponsibilityDisplayName(BigDecimal pMsiteID,
BigDecimal pRespID, BigDecimal pAppID, String pLanguageCode)
throws FrameworkException, SQLException
```

Given a minisite ID, responsibility and application ID combination, and language code, return the display name for the minisite and responsibility association. This method will return the display name if the combination of input values were found. It will return null, if it didn't find any match.

Parameters: pMsiteID - the minisite ID

pRespID - the responsibility ID

pAppID - the application ID

pLanguageCode - language code

Returns: Display name for the minisite responsibility combination

Throws: FrameworkException if error in getting a connection to the database

Throws: SQLException if error in communicating with the database

getName

```
public static String getName()
throws MinisiteNotExistsException, MinisiteException, SQLException,
FrameworkException
```

Get the name of the minisite, based on the language of the current session. If the name for the current session language is not available, then the name in the default language is returned. This API assumes that iStore cookies are set when calling the API.

Returns: the name of the minisite

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteException; occurs in case of application error

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getPriceListID

```
public static BigDecimal getPriceListID()
throws MinisiteNotExistsException, MinisiteException, SQLException,
FrameworkException
```

Get the price list ID based on the currency code and user type for the current session. If the currency code is not supported, price list ID based on the default currency is returned. This API assumes that iStore cookies are set when calling the API.

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteException; occurs in case of application error

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getRegisteredPriceListID

```
public static BigDecimal getRegisteredPriceListID()
throws MinisiteNotExistsException, MinisiteException, SQLException,
FrameworkException
```

Get the registered price list ID for the minisite. The price list ID returned is for the currency in the current session. This API assumes that iStore cookies are set when calling the API.

Returns: registered price list ID for the minisite

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteException; occurs in case of application error

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getRegisteredPriceListID

```
public static BigDecimal getRegisteredPriceListID(String pCurrencyCode)
```

```
throws MinisiteNotExistsException, MinisiteException, SQLException,
FrameworkException
```

Get the registered price list ID for the minisite for the given currency code. This API assumes that iStore cookies are set when calling the API.

Parameters: pCurrencyCode - currency code for which the registered price list needs to be returned.

Returns: registered price list ID for the minisite for the given currency code.

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteException; occurs in case of application error

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getRootSectionID

```
public static BigDecimal getRootSectionID()
throws MinisiteNotExistsException, SQLException, FrameworkException
```

Get the root section ID for the minisite. This API assumes that iStore cookies are set when calling the API.

Returns: root section ID for the minisite.

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getWalkinPriceListID

```
public static BigDecimal getWalkinPriceListID()
throws MinisiteNotExistsException, MinisiteException, SQLException,
FrameworkException
```

Get the walkin price list ID for the minisite. The price list ID returned is for the currency in the current session. This API assumes that iStore cookies are set when calling the API.

Returns: walkin price list ID for the minisite.

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteException; occurs in case of application error

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

getWalkinPriceListID

```
public static BigDecimal getWalkinPriceListID(String pCurrencyCode)
throws MinisiteNotExistsException, MinisiteException, SQLException,
FrameworkException
```

Get the walkin price list ID for the minisite for the given currency code. This API assumes that iStore cookies are set when calling the API.

Parameters: pCurrencyCode - currency code for which the walkin price list needs to be returned.

Returns: walkin price list ID for the minisite for the given currency code.

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteException; occurs in case of application error

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

isWalkinAllowed

```
public static boolean isWalkinAllowed()
throws MinisiteNotExistsException, SQLException, FrameworkException
```

Check if walkin is allowed in the minisite. This API assumes that iStore cookies are set when calling the API.

Returns: True if walkin is allowed for the minisite. Otherwise return false.

Throws: SQLException if error in communicating with the database

Throws: FrameworkException if error in getting a connection to the database

Throws: MinisiteNotExistsException if minisite cannot be determined from the request context or minisite does not exist

1.3 Class StoreProfile

```
java.lang.Object > oracle.apps.ibe.store.StoreProfile
```

public class **StoreProfile**

extends Object

StoreProfile contains all the utility functions to get the value of the various profiles in iStore.

1.3.1 Methods for Class StoreProfile

The following table is an index of Class StoreProfile methods:

Table 1–3 Method Index for Class StoreProfile

Method	Description
getCategorySetID	Get the value for iStore Category Set based on the profile IBE_CATEGORY_SET.
getItemsPerPage	Get the value for items per page based on the profile IBE_ITEMS_PER_PAGE.
getMaximumSearchResults	Get the value for maximum number of search results based on the profile IBE_RESULTS_IN_SEARCH.
getPriceRetrievalFlag	Get the price retrieval flag based on the profile IBE_PRICE_RETRIEVAL_FLAG.
getSearchLinesPerPage	Get the value for search lines per page based on the profile IBE_ITEMS_PER_PAGE.
getSectionsPerPage	Get the value for sections per page based on the profile IBE_SECTIONS_PER_PAGE.

getCategorySetID

```
public static int getCategorySetID()
throws MinisiteException
```

Get the value for iStore Category Set based on the profile IBE_CATEGORY_SET.

Returns: Returns the profile value for IBE_CATEGORY_SET if set, otherwise it throws an exception.

Throws: MinisiteException; occurs in case of application error

getItemsPerPage

```
public static int getItemsPerPage()
```

Get the value for items per page based on the profile IBE_ITEMS_PER_PAGE.

Returns: Returns the profile value for IBE_ITEMS_PER_PAGE if set, otherwise it returns the default value of 20.

getMaximumSearchResults

```
public static int getMaximumSearchResults()
```

Get the value for maximum number of search results based on the profile IBE_RESULTS_IN_SEARCH.

Returns: Returns the profile value for IBE_RESULTS_IN_SEARCH if set, otherwise it returns the default value of 200.

getPriceRetrievalFlag

```
public static boolean getPriceRetrievalFlag()
```

Get the price retrieval flag based on the profile IBE_PRICE_RETRIEVAL_FLAG.

Returns: Returns true if the profile value for IBE_PRICE_RETRIEVAL_FLAG is set to Y, otherwise it returns false.

getSearchLinesPerPage

```
public static int getSearchLinesPerPage()
```

Get the value for search lines per page based on the profile IBE_ITEMS_PER_PAGE.

Returns: Returns the profile value for IBE_SEARCH_LINES_PER_PAGE if set to a legal value, else default search lines per page is 20

getSectionsPerPage

```
public static int getSectionsPerPage()
```

Get the value for sections per page based on the profile IBE_SECTIONS_PER_PAGE.

Returns: Returns the profile value for IBE_SECTIONS_PER_PAGE if set, otherwise it returns the default value of 20.

2

Oracle iStore 11*i* Display Manager APIs

This chapter contains the following information on the Oracle iStore 11*i* Display Manager public class APIs in the package oracle.apps.ibe.displaymanager:

- Display Manager API Class Summary
- Class DisplayManager

2.1 Display Manager API Class Summary

APIs for the Oracle iStore 11*i* Display Manager are in the package oracle.apps.ibe.displaymanager. The table below describes the classes briefly.

Table 2–1 Display Manager Class Summary

Class Name	Description
Class DisplayManager	The DisplayManager Class is used to resolve the Logical to Physical mapping for Templates and Media.

2.2 Class DisplayManager

```
java.lang.Object > oracle.apps.ibe.displaymanager.DisplayManager
```

public final class **DisplayManager**

extends java.lang.Object

The DisplayManager Class is used to resolve the Logical to Physical mapping for Templates and Media. The class also provides methods to get the Physical Templates or Media given DisplayContext or MediaContext for an item or seciton. Site and Language are optional parameters, if not passed the values are derived from Request Context. Each of these methods return a Template or Media Object. The method getFileName() of Template or Media can be called to retrieve the physical file for a Template or Media.

2.2.1 Variables for Class DisplayManager

RCS_ID

```
public static final java.lang.String RCS_ID
```

RCS_ID_RECORDED

```
public static final boolean RCS_ID_RECORDED
```

2.2.2 Constructors for Class DisplayManager

DisplayManager

```
public DisplayManager()
```

2.2.3 Methods for Class DisplayManager

The following table is an index of Class DisplayManager methods:

Table 2–2 Method Index for Class DisplayManager

Method	Description
getItemMedia	Gets media for a product given a Media Context.The minisite ID and language are obtained from the RequestCtx instance for this request.
getItemTemplate	Gets the template for a product given DisplayContext. Also receives a particular site and language as parameters.
getMedia	Gets a media by its access name. Also receives a particular site and language as parameters.
getSectionMedia	Gets media for a given section and Media Context. Also receives a particular site and language as parameters.
getTemplate	Gets a template by its access name. Also receives a particular site and language as parameters.
getTextMedia	Retrieves back the text content from the text media by its access name. Also receives a particular site and language as parameters. Can also retrieve the text content from the text media by its Media object name.
getTextMediaOrFndMsg	Retrieves the text content from the given text media for the current minisite and language. If no applicable text content exists, the method retrieves the text content from the Oracle Applications Message Dictionary (FND) message that has the same name as the given text media. This method enables specification of text message content according to specialty stores.
getURL	Given a template name, gets the URL for the hyperlink. Also receives a query string as a parameter.
printTextMedia	Prints out the text content from the text media by its access name. Also receives a particular site and language as parameters. Can also print the text content from the text media by its Media object name.

getItemMedia

```
public static final oracle.apps.ibe.displaymanager.Media
getItemMedia(oracle.apps.ibe.catalog.Item product, java.lang.String
mediaContext)
throws java.sql.SQLException, oracle.apps.jtf.base.resources.FrameworkException
```

Gets media for a product given a Media Context.The minisite ID and language are obtained from the RequestCtx instance for this request.

Parameters: Item - product - the product for which media is needed

String - mediaContext - the media context

Returns: the Media object associated with the product and given media context Returns the default media if the required media is not found.

Throws: java.sql.SQLException - if a database error occurred

FrameworkException - if a connection could not be obtained

getItemTemplate

```
public static final oracle.apps.ibe.displaymanager.Template
getItemTemplate(oracle.apps.ibe.catalog.Item product, java.lang.String
displayContext)
throws java.sql.SQLException, oracle.apps.jtf.base.resources.FrameworkException
```

Gets the template for a product given DisplayContext. The minisite and language code are obtained from the Request Context instance for this request. If the template is not found for the product and display context, or if the display context itself is not found, the default product template is returned.

Parameters: Item - product - the product to be displayed

String - displayContext - the display context

Returns: the Template object associated with the product and displaycontext

Throws: java.sql.SQLException - if a database error occurred

FrameworkException - if a connection could not be obtained

getItemTemplate

```
public static final oracle.apps.ibe.displaymanager.Template
getItemTemplate(oracle.apps.ibe.catalog.Item product, java.lang.String
displayContext, int siteId, java.lang.String language)
throws java.sql.SQLException, oracle.apps.jtf.base.resources.FrameworkException
```

Gets the template for a product given a DisplayContext. The minisite and language code are obtained from the Request Context instance for this request.

Parameters: Item - product - the product which is to be displayed.

String - displayContext - the template displaycontext to be used for display

int - siteId - the minisite ID

String - language - the language code

Returns: the Template object associated with the product and displaycontext. If the required template is not found the default template is returned.

Throws: java.sql.SQLException - if a database error occurred

FrameworkException - if a connection could not be obtained

getMedia

```
public static final oracle.apps.ibe.displaymanager.Media
getMedia(java.lang.String mediaAccessName)
throws oracle.apps.ibe.displaymanager.MediaNotFoundException,
java.sql.SQLException, oracle.apps.jtf.base.resources.FrameworkException
```

Gets a media by its access name. The minisite ID and language code are obtained from the RequestCtx instance for this request.

Parameters: String - mediaAccessName - the access name of the template

Returns: the Media object associated with the access name

Throws: oracle.apps.ibe.displaymanager.MediaNotFoundException - if the media is not found

java.sql.SQLException - if a database error occurred

FrameworkException - if a connection could not be obtained

getMedia

```
public static final oracle.apps.ibe.displaymanager.Media
getMedia(java.lang.String mediaAccessName, int siteId, java.lang.String
language)
throws oracle.apps.ibe.displaymanager.MediaNotFoundException,
java.sql.SQLException, oracle.apps.jtf.base.resources.FrameworkException
```

Gets a media by its access name for a particular site and language.

Parameters: String - mediaAccessName - the access name of the media

int - siteId - the minisite ID

String - language - the language code

Returns: the Media object associated with the access name

Throws: oracle.apps.ibe.displaymanager.MediaNotFoundException - if the media is not found

java.sql.SQLException - if a database error occurred

FrameworkException - if a connection could not be obtained

getSectionMedia

```
public static final oracle.apps.ibe.displaymanager.Media getSectionMedia(int
sectionId, java.lang.String mediaContext)
throws java.sql.SQLException, oracle.apps.jtf.base.resources.FrameworkException
```

Gets media for a given section and Media Context. The minisite ID and language code are obtained from the RequestCtx instance for this request

Parameters: int - sectionId - the section ID

String - mediaContext - the media context

Returns: the Media object associated with the section and media context Returns the default media if the required media is not found.

Throws: java.sql.SQLException - if a database error occurred

FrameworkException - if a connection could not be obtained

getSectionMedia

```
public static final oracle.apps.ibe.displaymanager.Media getSectionMedia(int
sectionId, java.lang.String mediaContext, int siteId, java.lang.String language)
throws java.sql.SQLException, oracle.apps.jtf.base.resources.FrameworkException
```

Gets media for a given section and Media Context, in a particular site and language.

Parameters: int - sectionId - the section ID

String - mediaContext - the media context

int - siteId - the minisite ID

String - language - the language code

Returns: the Media object associated with the section and media context Returns the default media if the required media is not found.

Throws: java.sql.SQLException - if a database error occurred

FrameworkException - if a connection could not be obtained

getTemplate

```
public static final oracle.apps.ibe.displaymanager.Template
getTemplate(java.lang.String accessName)
throws oracle.apps.ibe.displaymanager.TemplateNotFoundException,
java.sql.SQLException,  oracle.apps.jtf.base.resources.FrameworkException
```

Gets a template by its access name. The minisite ID and language are obtained from the RequestCtx instance for this request.

Parameters: String - accessName - the access name of the template

Returns: the Template object associated with the access name

Throws: oracle.apps.ibe.displaymanager.TemplateNotFoundException - if the template is not found

java.sql.SQLException - if a database error occurred

FrameworkException - if a connection could not be obtained

getTemplate

```
public static final oracle.apps.ibe.displaymanager.Template
getTemplate(java.lang.String accessName, int siteId, java.lang.String language)
throws oracle.apps.ibe.displaymanager.TemplateNotFoundException,
java.sql.SQLException, oracle.apps.jtf.base.resources.FrameworkException
```

Gets a template by its access name for a particular minisite and language.

Parameters: String - accessName - the access name of the template

int - siteId - the minisite ID

String - language - the language code in which the template is needed

Returns: the Template object associated with the access name

Throws: oracle.apps.ibe.displaymanager.TemplateNotFoundException - if the template is not found

java.sql.SQLException - if a database error occurred

FrameworkException - if a connection could not be obtained

getTextMedia

```
public static java.lang.String getTextMedia(java.lang.String mediaAccessName)
throws oracle.apps.ibe.displaymanager.MediaException
```

Retrieves back the text content from the text media by its access name. The minisite ID and language code are obtained from the RequestCtx instance for this request. Text media are any plain text files stored in database or the file system.

Parameters: String - mediaAccessName - the access name of the media

Returns: the content of the text media

Throws: oracle.apps.ibe.displaymanager.MediaException - if any errors

getTextMedia

```
public static java.lang.String getTextMedia(java.lang.String mediaAccessName,
int siteId, java.lang.String language)
throws oracle.apps.ibe.displaymanager.MediaException
```

Retrieves back the text content from the text media by its access name for a particular site and language. Text media are any plain text files stored in database or the file system.

Parameters: String - mediaAccessName - the access name of the media

int - siteId - the minisite ID

String - language - the language code

Returns: the content of the text media

Throws: oracle.apps.ibe.displaymanager.MediaException - if any errors

getTextMedia

```
public static java.lang.String getTextMedia(oracle.apps.ibe.displaymanager.Media
m)
throws oracle.apps.ibe.displaymanager.MediaException
```

Retrieves back the text content from the given text media.

Parameters: m - the given text media object

Returns: the content of the text media

Throws: oracle.apps.ibe.displaymanager.MediaException - if any errors

getTextMediaOrFndMsg

```
public static String getTextMediaOrFndMsg(String pMediaAccessName)
throws MediaException
```

Retrieves the text content from the given text media for the current minisite and language. If no applicable text content exists, the method retrieves the text content from the Oracle Applications Message Dictionary (FND) message that has the same name as the given text media. This method enables specification of text message content according to specialty stores.

Parameters: pMediaAccessName - the multimedia object's programmatic access name

Returns: the content of the text media, or if there is none, the content of the FND message

Throws: oracle.apps.ibe.displaymanager.MediaException - if any errors

getURL

```
public static java.lang.String getURL(java.lang.String accessName)
throws java.sql.SQLException, oracle.apps.jtf.base.resources.FrameworkException
```

Given a template name, gets the URL for the hyperlink. To be called from JavaServer Pages™ (JSP™) to get URLs.

Parameters: String - accessName - the access name of the template

Returns: URL

Throws: java.sql.SQLException - if a database error occurred

FrameworkException - other errors

getURL

```
public static java.lang.String getURL(java.lang.String accessName,
java.lang.String queryString)
throws java.sql.SQLException, oracle.apps.jtf.base.resources.FrameworkException
```

Given a template name, gets the URL for the hyperlink. The query string is added to the URL. To be called from JSPs to get URLs.

Parameters: String - accessName - the access name of the template

String - queryString - (example: foo1=bar1&foo2=bar2)

Returns: URL

Throws: java.sql.SQLException - if a database error occurred

FrameworkException - other errors

printTextMedia

```
public static void printTextMedia(java.lang.String mediaAccessName,
oracle.apps.ibe.displaymanager.JspWriter out)
throws oracle.apps.ibe.displaymanager.MediaException
```

Prints out the text content from the text media by its access name. The minisite ID and language code are obtained from the RequestCtx instance for this request. Text media are any plain text files stored in database or the file system.

Parameters: String - mediaAccessName - the access name of the media

JspWriter - out - the output writer

Throws: oracle.apps.ibe.displaymanager.MediaException - if any errors

printTextMedia

```
public static void printTextMedia(java.lang.String mediaAccessName, int siteId,
java.lang.String language, oracle.apps.ibe.displaymanager.JspWriter out)
throws oracle.apps.ibe.displaymanager.MediaException
```

Prints out the text content from the text media by its access name for a particular site and language. Text media are any plain text files stored in database or the file system.

Parameters: String - mediaAccessName - the access name of the media

int - siteId - the minisite ID

String - language - the language code

JspWriter - out - the output writer

Throws: oracle.apps.ibe.displaymanager.MediaException - if any errors

printTextMedia

```
public static void printTextMedia(oracle.apps.ibe.displaymanager.Media m,
oracle.apps.ibe.displaymanager.JspWriter out)
throws oracle.apps.ibe.displaymanager.MediaException
```

Prints out the text content from the given text media.

Parameters: m - the given text media object

out - the output writer

Throws: oracle.apps.ibe.displaymanager.MediaException - if any errors

3

Oracle iStore 11*i* Catalog APIs

This chapter contains the following information about the Oracle iStore 11*i* Catalog public class APIs in the package oracle.apps.ibe.catalog:

- Catalog API Class Summary
- Class Item
- Class ItemFlexfield
- Class PriceObject
- Class Section
- Exception Classes

3.1 Catalog API Class Summary

APIs for the Oracle iStore 11*i* Catalog are in the package oracle.apps.ibe.catalog. The table below describes the classes briefly.

Table 3–1 Class Summary for the Package oracle.apps.ibe.catalog

Class	Description
Class Item	The Item object catches the selling side information for an item in Oracle Inventory. It is the entity that customers can add into the shopping cart from the Web store. The Item object is used to retrieve basic item attributes stored in MTL_SYSTEM_ITEMS_VL, such as part number, description, and long description. It is also used to retrieve the template and multimedia associated with the item for a specific display context, the list of units of measure for the item, the prices defined for the item, whether the item is configurable, the list of related items, the list of related sections, and the item flexfields.
Class ItemFlexfield	The ItemFlexfield object contains the segment information for an item flexfield segment. It is used to retrieve the name, prompt, value, and database column name for an item flexfield segment. This object is returned by the getFlexfields() APIs in the Item class.
Class PriceObject	The PriceObject contains pricing information retrieved for an item. It is used to retrieve list price and selling price. It also provides the functionality to format a price based on currency. This object is returned by the getListAndBestPrices() APIs in the Item class.

Table 3–1 Class Summary for the Package oracle.apps.ibe.catalog (Cont.)

Class	Description
Class Section	The Section object is the building block of the display hierarchy. The display hierarchy is a tree structure which defines the possible navigational paths for the store. There are two types of sections: FEATURED and NAVIGATIONAL. Featured sections cannot contain subsections and are not displayed in the browse hierarchy of the store. Navigational sections can contain subsections and are displayed in the browse hierarchy of the store. Each section (except the hierarchy root) has one parent section and one or more subsections. A section without subsections is a leaf section. Leaf sections are the only sections that can contain items. The Section object is used to retrieve basic section attributes stored in jtf_dsp_sections_vl (such as display name, description, long description, etc.). It is also used to retrieve the template associated with the section, the multimedia associated with the section for a specific display context, the list of supersections, the list of subsections of a certain type (FEATURED or NAVIGATIONAL), the list of items, the list of sibling sections, and the list of related sections.

3.2 Class Item

`java.lang.Object > oracle.apps.ibe.catalog.Item`

public class **Item**

extends Object

The Item object stores the selling side attributes for an Oracle Inventory item in MTL_SYSTEM_ITEMS_VL. It is also used to retrieve the template and multimedia associated to the item for a specific display context, the list of units of measure for the item, the prices defined for the item, whether the item is configurable, the list of related items, the list of related sections, and the item flexfields.

An Item object is built by using the load() APIs, passing in item ID or part number. The item ID typically comes from the leaf section when browsing through the hierarchy. When retrieving item prices, it is recommended that you use the getListAndBestPrices() APIs, which return both list price and best price for all UOMs available to an item. If only the price for a particular UOM is needed, you may prefer to use the getBestPrices() and getListPrices() APIs. For all price APIs, if price list is null (either a null parameter is passed, or the minisite set up for the price list is null), the current user information (party ID and account ID) are used to determine the price list ID.

3.2.1 Variables for Class Item

PUBLISHED

```
public static final String PUBLISHED
```

SHALLOW

```
public static final int SHALLOW
```

SHALLOW is a constant passed into Item load APIs to request an Item shallow load, which will load the following item information from Oracle Inventory:

- BOM_ENABLED_FLAG
- ORDERABLE_ON_WEB_FLAG
- BACK_ORDERABLE_FLAG
- PRIMARY_UNIT_OF_MEASURE
- UNIT_OF_MEASURE_TL
- PRIMARY_UOM_CODE
- ITEM_TYPE
- DESCRIPTION
- LONG_DESCRIPTION
- BOM_ITEM_TYPE
- CONCATENATED_SEGMENTS (part number)
- INVENTORY_ITEM_ID

If other information other than the above is requested from a shallow loaded item, the item will automatically be DEEP loaded, after which all the information will be available.

DEEP

```
public static final int DEEP
```

DEEP is a constant passed into Item load APIs to request an Item deep load, which will load all item attributes.

MODEL

```
public static final int MODEL
```

Constant to identify a BOM_ITEM_TYPE of Model

OPTION_CLASS

```
public static final int OPTION_CLASS
```

Constant to identify a BOM_ITEM_TYPE of Option Class

3.2.2 Methods for Class Item

The following table is an index of Class Item methods:

Table 3–2 Method Index for Class Item

Method	Description
checkIfValid	Retrieves whether the item with the ID(s) passed in as parameter is a valid item that should be displayed in the Web store
getATPFlag	Retrieves whether ATP check must be performed on the item
getAttributeCategory	Retrieves attribute_category column of the item
getAttributeColumn	Retrieves a column from attribute1-15 in MTL_SYSTEM_ITEMS_B table
getBestPrice	Retrieves the best price for the item, based on UOM codes
getBOMComponentIDs	Retrieves a BOM item’s component items from the BOM structure
getBOMComponents	Retrieves a BOM item’s component items from the BOM structure based on the user's organization
getBomItemType	Retrieves item's BOM item type
getColumnValue	Retrieves the value of a column from MTL_SYSTEM_ITEMS_VL for this item
getDescription	Retrieves the item's description column based on the user's language
getFixedOrderQty	Retrieves the item's fixed order quantity
getFlexfields	Retrieves the flexfield segments in the flexfield MTL_SYSTEM_ITEMS

Table 3–2 Method Index for Class Item (Cont.)

Method	Description
getGlobalAttributeCategory	Retrieves global_attribute_category column of the item
getGlobalAttributeColumn	Retrieves a column from Global_Attribute1-10 in MTL_ SYSTEM_ITEMS
getItemID	Retrieves item ID
getItemType	Retrieves item's user defined item type
getListAndBestPrices	Retrieves the list and best prices for each UOM of each item in the array passed in as parameter based on the minisite's price list ID
getListPrice	Retrieves the list price of the item for the primary UOM code
getLongDescription	Retrieves the item's long description column
getMaxOrderQty	Retrieves the item's maximum order quantity
getMediaFileName	Retrieves the file name of the physical media associated with this item for a particular display context
getMinOrderQty	Retrieves the item's minimum order quantity
getPartNumber	Retrieves part number
getPrimaryUOM	Retrieves the item's primary UOM, based on the user's language
getPrimaryUOMCode	Retrieves the item's primary UOM code
getRelatedItemIDs	Retrieves the IDs of items related to this item by the relationship code passed in as parameter
getRelatedItems	Retrieves the items related to this item by the relationship code passed in as parameter
getRelatedPrice	Retrieves the price of an item whose price is based on this item's price
getRelatedPrices	Retrieves the prices of items whose price is based on this item's price
getRelatedSectionIDs	Retrieves IDs of sections related to this item by the relationship code passed in as pararmeter
getSegmentColumn	Retrieves the value of column from Segment1-20 in MTL_ SYSTEM_ITEMS_VL
getSrvcDuration	Retrieves the default service duration
getSrvcPeriod	Retrieves the item's period for default service duration

Table 3–2 Method Index for Class Item (Cont.)

Method	Description
getSrvcStartDelay	Retrieves the number of days after shipment that service begins
getTemplateFileName	Retrieves the file name of the physical template associated with this item for a particular display context
getUOM	Retrieves the translated UOM based on the user's language for the UOM code passed in as parameter
getUOMCodes	If the profile option IBE: Retrieve All Units of Measure for an Item is set to 'Yes' or does not have a value, retrieves all the UOM codes defined for the item. If the profile option IBE: Retrieve All Units of Measure for an Item is set to 'No', retrieves the primary UOM code.
getUOMs	Retrieves the UOMs, based on the user's language
isBackOrderable	Retrieves whether the item can be back ordered
isBomEnabled	Retrieves whether the item is BOM enabled
isConfigurable	Retrieves whether the item can be configured
isCouponExempt	Retrieves whether the item is coupon exempt
isDownloadable	Retrieves whether the item is downloadable
isElectronic	Retrieves whether the item is electronic
isOrderable	Retrieves whether the item is orderable via Web
isReturnable	Retrieves whether the item is returnable
isService	Retrieves whether the item is a service item
isServiceable	Retrieves whether the item is serviceable
isShippable	Retrieves whether the item is shippable
isTaxable	Retrieves whether the item is taxable
isVolDiscountExempt	Retrieves whether the item is volume discount exempt
load	Loads the item with the parameters passed in
validateQuantity	Determines whether the input quantity is valid for the item

checkIfValid

```
public static boolean checkIfValid(BigDecimal itmid)
throws SQLException, FrameworkException
```

Check if the item with the ID passed in as parameter is a valid item that should be displayed in the Web store. A valid item must be effective, published, and exist in the current inventory validation organization. This API always queries the database.

Parameters: itmid - ID of the item to be validated

Returns: boolean - true if the item is valid and should be displayed, false otherwise

checkIfValid

```
public static boolean[] checkIfValid(BigDecimal itmids[])
throws SQLException, FrameworkException
```

Check if the items with the IDs passed in as parameter are valid items that should be displayed in the Web store. This API always queries the database.

Parameters: itmids - IDs of the items to be validated

Returns: boolean[] - array containing whether an item is valid and should be displayed. If itmids[i] is a valid item, boolean[i] is true. Otherwise, boolean[i] is false. Returns boolean[] of size 0 if itmids is null or itmids.length is 0.

getATPFlag

```
public boolean getATPFlag()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves whether ATP check must be performed on the item

Returns: true if ATP check must be performed, false otherwise

getAttributeCategory

```
public String getAttributeCategory()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves the attribute_category column of the item

Returns: attribute category of the item

getAttributeColumn

```
public String getAttributeColumn(int k)
throws SQLException, FrameworkException, InvalidColumnNumberException,
ItemNotFoundException
```

Retrieves a column from attribute1-15 in MTL_SYSTEM_ITEMS_B table

Parameters: k - int representing which attribute column to return

Returns: attribute value in column MTL_SYSTEM_ITEMS_VL.ATTRIBUTEk

getBestPrice

```
public BigDecimal getBestPrice()
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the best price for the item, based on primary UOM code. Uses minisite to determine price list ID. If price list ID is null, uses party ID and account ID to determine which price list to use.

Returns: item’s best price for the primary UOM code and minisite price list ID.

getBestPrice

```
public BigDecimal getBestPrice(String uomCode)
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the best price for the item, based on UOM code passed in as parameter. Uses minisite to determine price list ID. If price list ID is null, uses party ID and account ID to determine which price list to use.

Parameters: uomCode - UOM code used to retrieve the price

Returns: item's best price for uomCode passed in as parameter

getBestPrice

```
public BigDecimal getBestPrice(BigDecimal priceListId)
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the best price of the item, based on primary UOM code and price list ID passed in as parameter.

Parameters: priceListId - price list ID used to retrieve the price

Returns: item’s best price for primary UOM code and price list ID passed in as parameter.

getBestPrice

```
public BigDecimal getBestPrice(String uomCode, BigDecimal priceListId)
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the best price for the item, based on UOM code and price list ID passed in as parameter.

Parameters: uomCode - UOM code used to retrieve the price

priceListId - price list ID used to retrieve the price

Returns: item's best price for uomCode and price list passed in as parameter

getBestPrice

```
public BigDecimal getBestPrice(BigDecimal partyId, BigDecimal accountId)
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the best price of the item based on the customer for the primary UOM code. Minisite's price list ID will be used. If minisite's price list ID is null, uses customer information to determine price list.

Parameters: partyId - customer's partyId

accountId - customer's accountId

Returns: item's best price based on customer for the primary UOM code

getBestPrice

```
public BigDecimal getBestPrice(BigDecimal priceListId, BigDecimal partyId,
BigDecimal accountId)
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the best price of the item based on price list and customer for the primary UOM code. If price list ID is null, uses customer information to determine which price list to use. Otherwise, uses the price list passed in as parameter.

Parameters: priceListId - price list ID to use for pricing

partyId - customer's partyId

accountId - customer's accountId

Returns: item's best price based on price list and customer for the primary UOM code

getBestPrice

```
public BigDecimal getBestPrice(String uomCode, BigDecimal partyId, BigDecimal
accountId)
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the best price of the item based on the customer for the UOM code passed in as parameter. Minisite's price list ID will be used. If price list ID is null, Customer information will be used to determine the price list to use.

Parameters: uomCode - UOM code used to retrieve the price

partyId - customer's partyId

accountId - customer's accountId

Returns: item's best price based on customer

getBestPrice

```
public BigDecimal getBestPrice(String uomCode, BigDecimal priceListId,
BigDecimal partyId, BigDecimal accountId)
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the best price of the item based on the price list and customer for the UOM code passed in as parameter. If price list ID is null, customer information will be used to determine which price list to use. Otherwise, uses the price list passed in as parameter.

Parameters: uomCode - UOM code used to retrieve the price

priceListId - price list to use for pricing

partyId - customer's partyId

accountId - customer's accountId

Returns: item's best price based on customer

getBOMComponentIDs

```
public int[] getBOMComponentIDs()
throws FrameworkException, SQLException
```

If the item is a BOM item, this method retrieves its component items from the BOM structure.

Returns: array of item IDs which are the first level components of the current item. Array of size 0 if the current item is not a BOM item or has no active BOM components

getBOMComponents

```
public Item[] getBOMComponents()
throws SQLException, FrameworkException
```

If the item is a BOM item, this method retrieves its component items from the BOM structure based on the user's organization.

Returns: array of items which are the first level components of the current item, array of size 0 if the current item is not a BOM item or has no active BOM components

getBomItemType

```
public int getBomItemType()
```

Retrieves the item's BOM item type

Returns: item's BOM item type

getColumnValue

```
public String getColumnValue(String colName)
throws SQLException, FrameworkException
```

Retrieves the value of a column from MTL_SYSTEM_ITEMS_VL for this item

Parameters: colName - name of the column in MTL_SYSTEM_ITEMS_VL whose value will be retrieved

Returns: value of a column from MTL_SYSTEM_ITEMS_VL for this item

getDescription

```
public String getDescription()
throws SQLException, FrameworkException, ItemNotFoundException
```

Retrieves the item's description column based on the user's language

Returns: display name of the item based on the user's language

getFixedOrderQty

```
public int getFixedOrderQty()
throws SQLException, FrameworkException, ItemNotFoundException
```

Retrieves the item's fixed order quantity

Returns: item's fixed order quantity; -1 if NULL in MTL_SYSTEM_ITEMS

getFlexfields

```
public ItemFlexfield[] getFlexfields()
throws FrameworkException,SQLException
```

Retrieves the flexfield segments in the flexfield "MTL_SYSTEM_ITEMS." Flexfield segment information (name, prompt, value, database column name) can be retrieved from the ItemFlexfield object using the methods getName(), getPrompt(), getValue(), and getDbColumnName().

Returns: array of ItemFlexfield containing flexfield segment information.

getFlexfields

```
public ItemFlexfield[] getFlexfields(String application, String flexfieldName)
throws FrameworkException,SQLException
```

Retrieves the flexfield segments in the flexfield with the application short name and flexfield name passed in as parameter. Flexfield segment information (name, prompt, value, database column name) can be retrieved from the ItemFlexfield object using the methods getName(), getPrompt(), getValue(), and getDbColumnName().

Parameters: application - short name of the application module to which the flexfield belongs; for example, “INV”

flexfieldName - name of the flexfield; for example, “MTL_SYSTEM_ITEMS”

Returns: array of ItemFlexfield containing flexfield segment information.

getGlobalAttributeCategory

```
public String getGlobalAttributeCategory()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves global_attribute_category column of the the item

Returns: global attribute category of the item

getGlobalAttributeColumn

```
public String getGlobalAttributeColumn(int k)
throws FrameworkException, SQLException, InvalidColumnNumberException,
ItemNotFoundException
```

Retrieves a column from Global_Attribute1-10 in MTL_SYSTEM_ITEMS

Parameters: k - int representing which attribute to return

Returns: attribute value in column MTL_SYSTEM_ITEMS.GLOBAL_ATTRIBUTEk

getItemID

```
public int getItemID()
```

Retrieves item ID

Returns: the ID of the item

getItemType

```
public String getItemType()
```

Retrieves the item's user defined item type

Returns: item's item type

getListAndBestPrices

```
public static Vector[] getListAndBestPrices(Item itms[])
throws FrameworkException, SQLException, CatalogException
```

Retrieves the list and best prices for each UOM of each item in the array passed in as parameter based on the minisite's price list ID. If there is no price list ID for the minisite, passes party ID and account ID to the pricing engine to determine the price list. Prices will be returned in a Vector[]. The size of the Vector[] will be the same as the size of the Item[] passed in as parameter. Vector[i] will contain a vector of PriceObject for Item[i]. Each PriceObject corresponds to the price based on a UOM code. The PriceObjects in Vector[i] will be ordered in the same order as the UOM codes of Item[i]. To ensure this ordering, iterate through each Vector using elementAt(n) since Enumeration does not guarantee the order in which the elements are returned. The Vector will be trimmed to size to ensure that that there will be a PriceObject for each n from 0 to size()-1. To obtain the list and best price call, getListPrice, getBestPrice() on each PriceObject.

Parameters: itms - items whose list and best prices will be retrieved

Returns: Vector[] containing list and best prices for the UOMs of the corresponding item. If Item[] is null or the size of Item[] is 0, returns Vector[] of size 0. If Item[I] has no UOM codes, Vector[I] will be an empty Vector.

getListAndBestPrices

```
public static Vector[] getListAndBestPrices(Item itms[], BigDecimal priceListId)
throws FrameworkException, SQLException, CatalogException
```

Retrieves the list and best prices for each UOM of each item in the array passed in as parameter, based on the price list ID. If price list ID is not null, retrieves price based only on price list ID and caches the prices. Otherwise, uses party ID and account ID to determine price list. Prices will be returned in a Vector[]. The size of the Vector[] will be the same as the size of the Item[] passed in as parameter. Vector[i] will contain a vector of PriceObject for Item[i]. Each PriceObject corresponds to the price based on a UOM code. The PriceObjects in Vector[i] will be ordered in the same order as the UOM codes of Item[i]. To ensure this ordering, iterate through each Vector using elementAt(n) since Enumeration does not guarantee the order in which the elements are returned. The Vector will be trimmed to size to ensure that that there will be PriceObjects for each n from 0 to size()-1. To obtain the list and best price call getBestPrice(), getListPrice() on each PriceObject.

Parameters: itms - items whose list and best prices will be retrieved

priceListId - price list ID used to retrieve price

Returns: Vector[] containing list and best prices for the UOMs of the corresponding item. If Item[] is null or the size of Item[] is 0, returns Vector[] of size 0. If Item[i] has no UOM codes, Vector[i] will be an empty Vector.

getListAndBestPrices

```
public static Vector[] getListAndBestPrices(Item itms[], BigDecimal partyId,
BigDecimal accountId)
throws FrameworkException, SQLException, CatalogException
```

Retrieves the best and list prices for each UOM of each item in the array passed in as parameter, based on the minisite’s price list ID and customer. List and best prices will be returned in a Vector[]. The size of the Vector[] will be the same as the size of the Item[] passed in as parameter. Vector[i] will contain a vector of PriceObject for Item[i]. Each PriceObject corresponds to the price based on a UOM code. The PriceObjects in Vector[i] will be ordered in the same order as the UOM codes of Item[i]. To ensure this ordering, iterate through each Vector using elementAt(n) since Enumeration does not guarantee the order in which the elements are returned. The Vector will be trimmed to size to ensure that that there will be PriceObjects for each n from 0 to size()-1. To obtain the list price and best price call getListPrice(), getBestPrice() on each PriceObject.

Parameters: itms - items whose list and best prices will be retrieved

priceListId - price list ID used to retrieve price

Returns: Vector[] containing list and best prices for the UOMs of the corresponding item. If Item[] is null or the size of Item[] is 0, returns Vector[] of size 0. If Item[I] has no UOM codes, Vector[I] will be an empty Vector.

getListAndBestPrices

```
public static Vector[] getListAndBestPrices(Item itms[], BigDecimal priceListId,
BigDecimal partyId, BigDecimal accountId)
throws FrameworkException, SQLException, CatalogException
```

Retrieves the best and list prices for each UOM of each item in the array passed in as parameter, based on the price list and customer. If price list ID is null, uses the customer information to determine which price list to use. Otherwise, uses the price list passed in as parameter. List and best prices will be returned in a Vector[]. The size of the Vector[] will be the same as the size of the Item[] passed in as parameter. Vector[i] will contain a vector of PriceObject for Item[i]. Each PriceObject corresponds to the price based on a UOM code. The PriceObjects in Vector[i] will be ordered in the same order as the UOM codes of Item[i]. To ensure this ordering, iterate through each Vector using elementAt(n) since Enumeration does not guarantee the order in which the elements are returned. The Vector will be trimmed to size to ensure that that there will be PriceObjects for each n from 0 to size()-1. To obtain the list price and best price call getListPrice(), getBestPrice() on each PriceObject.

Parameters: itms - items whose best prices will be retrieved

priceListId - price list used for pricing

partyId - customer's party ID

accountId - customer's account ID

Returns: Vector[ ] containing list and best prices for the UOMs of the corresponding item. If Item[]is null or the size of Item[ ] is 0, returns Vector[]of size 0. If Item[i] has no UOM codes, Vector[i] will be an empty Vector.

getListPrice

```
public BigDecimal getListPrice()
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the list price of the item for the primary UOM code

Returns: item's list price for the primary UOM code

getListPrice

```
public BigDecimal getListPrice(String uomCode)
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the list price of the item for the UOM code passed in as parameter. Uses minisite to determine price list ID.

Parameters: uomCode - UOM code used to retrieve price

Returns: item's list price for the UOM code passed in as parameter

getListPrice

```
public BigDecimal getListPrice(BigDecimal priceListId)
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the list price of the item for the primary UOM code based on price list ID passed in as parameter

Parameters: priceListId - price list ID used to retrieve price

Returns: item's list price for the primary UOM code

getListPrice

```
public BigDecimal getListPrice(String uomCode, BigDecimal priceListId)
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the list price of the item for the UOM code passed in as parameter. Uses price list ID passed in as parameter.

Parameters: uomCode - UOM code used to retrieve price

priceListId - price list ID used to retrieve price

Returns: item's list price for the UOM code passed in as parameter

getListPrice

```
public BigDecimal getListPrice(BigDecimal partyId, BigDecimal accountId)
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the list price of the item based on the customer for the item's primary uomCode

Parameters: partyId - customer's partyId

accountId - customer's accountId

Returns: item's list price based on customer

getListPrice

```
public BigDecimal getListPrice(String uomCode, BigDecimal partyId, BigDecimal
accountId)
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the list price of the item based on minisite's price list ID and the customer for the uomCode passed in as parameter

Parameters: uomCode - UOM code used to get the price

partyId - customer's partyId

accountId - customer's accountId

Returns: item's list price based on customer

getListPrice

```
public BigDecimal getListPrice(BigDecimal priceListId, BigDecimal partyId,
BigDecimal accountId)
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the list price of the item based on price list and customer for the item's primary uomCode. If price list ID is null, uses the customer information to determine which price list to use. Otherwise, uses the price list passed in as parameter.

Parameters: priceListId - price list ID used for pricing

partyId - customer's partyId

accountId - customer's accountId

Returns: item's list price based on customer

getListPrice

```
public BigDecimal getListPrice(String uomCode, BigDecimal priceListId,
BigDecimal partyId, BigDecimal accountId)
throws SQLException, FrameworkException, PriceNotFoundException
```

Retrieves the list price of the item based on price list and customer for the uomCode passed in as parameter. If price list ID is null, uses the customer information to determine which price list to use. Otherwise, uses the price list passed in as parameter.

Parameters: uomCode - UOM code used to get the price

priceListId - price list used for pricing

partyId - customer's partyId

accountId - customer's accountId

Returns: item's list price based on customer

getLongDescription

```
public String getLongDescription()
throws SQLException, FrameworkException, ItemNotFoundException
```

Retrieves the item's long description column

Returns: longDescription of the item based on the user's language

getMaxOrderQty

```
public int getMaxOrderQty()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves item's maximum order quantity

Returns: item's maximum order quantity; -1 if NULL in MTL_SYSTEM_ITEMS

getMediaFileName

```
public String getMediaFileName(String dispCtx)
throws FrameworkException, SQLException
```

Retrieves the file name of the physical media associated with this item for a particular display context

Parameters: dispCtx - display context for the media; for example, STORE_
PRODUCT_SMALL_IMAGE or STORE_PRODUCT_LARGE_IMAGE

Returns: file name of the physical media. If the required media is not found, returns null.

getMinOrderQty

```
public int getMinOrderQty()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves item's minimum order quantity

Returns: item's minimum order quantity; -1 if NULL in MTL_SYSTEM_ITEMS

getPartNumber

```
public String getPartNumber()
```

Retrieves part number

Returns: the part number of the item

getPrimaryUOM

```
public String getPrimaryUOM()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves the item's primary UOM, based on the user's language

Returns: item's UOM based on the user's language

getPrimaryUOMCode

```
public String getPrimaryUOMCode()
```

Retrieves primary UOM code for the item

Returns: item's primary UOM code

getRelatedItemIDs

```
public int[] getRelatedItemIDs(String relationshipCode)
throws SQLException, FrameworkException, ItemNotFoundException, CatalogException
```

Retrieves IDs of items related to this item by the relationship code passed in as parameter. This method should only be used for relationships that are not defined by a SQL rule and that do not require bind arguments.

Parameters: relationshipCode - specifies the type of relationship; for example, "SUBSTITUTE"

Returns: IDs of items related to this item by the relationship code passed in as parameter

getRelatedItemIDs

```
public int[] getRelatedItemIDs(String relationshipCode, boolean isSQLRule)
throws SQLException, FrameworkException, CatalogException, ItemNotFoundException
```

Retrieves IDs of items related to this item by the relationship code passed in as parameter. This method should only be used for relationships that do not require bind arguments.

Parameters: relationshipCode - specifies the type of relationship; for example, "SUBSTITUTE"

isSQLRule - whether the relationship is defined by a SQL rule

Returns: IDs of items related to this item by the relationship code passed in as parameter

getRelatedItemIDs

```
public int[] getRelatedItemIDs(String relationshipCode, boolean isSQLRule,
String bindArgs[])
throws SQLException, FrameworkException, CatalogException, ItemNotFoundException
```

Retrieves IDs of items related to this item by the relationship code passed in as parameter.

Parameters: relationshipCode - specifies the type of relationship; for example, "SUBSTITUTE"

isSQLRule - whether the relationship is defined by a SQL rule

bindArgs - array of Strings containing bind arguments, used in relationships defined using SQL rules with bind arguments

Returns: IDs of items related to this item by the relationship code passed in as parameter

getRelatedItems

```
public Item[] getRelatedItems(String relationshipCode)
throws SQLException, FrameworkException, CatalogException, ItemNotFoundException
```

Retrieves items related to this item by the relationship code passed in as parameter. This method should only be used for relationships that are not defined by a SQL rule and that do not require bind arguments.

Parameters: relationshipCode - specifies the type of relationship; for example, "SUBSTITUTE"

Returns: array of items related to this item by the relationship code passed in as parameter

getRelatedItems

```
public Item[] getRelatedItems(String relationshipCode, int front, int end)
throws SQLException, FrameworkException, CatalogException, ItemNotFoundException
```

Retrieves subarray of items related to this item by the relationship code passed in as parameter. This method should only be used for relationships that are not defined by a SQL rule and that do not require bind arguments. The subarray will contain items whose position in the related item ID array is between the front and end indexes passed in as parameter. If the end index is larger than the number of related items, returns the items from the front index to the end of the list of items.

Parameters: relationshipCode - specifies the type of relationship; for example, "SUBSTITUTE"

front - beginning index of items to load. Indexing starts at 0.

end - ending index of items to load

Returns: subarray of items related to this item by the relationship code passed in as parameter

getRelatedItems

```
public Item[] getRelatedItems(String relationshipCode, boolean isSQLRule)
throws SQLException, FrameworkException, CatalogException, ItemNotFoundException
```

Retrieves items related to this item by the relationship code passed in as parameter. This method should only be used for relationships that do not require bind arguments.

Parameters: relationshipCode - specifies the type of relationship; for example, "SUBSTITUTE"

isSQLRule - whether the relationship is defined by a SQL rule

Returns: items related to this item by the relationship code passed in as parameter

getRelatedItems

```
public Item[] getRelatedItems(String relationshipCode, boolean isSQLRule, int
front, int end)
throws SQLException, FrameworkException, CatalogException, ItemNotFoundException
```

Retrieves subarray of items related to this item by the relationship code passed in as parameter. This method should only be used for relationships that do not require bind arguments. The subarray will contain items whose position in the related item

ID array is between the front and end indexes passed in as parameter. If the end index is larger than the number of related items, returns the items from the front index to the end of the list of items.

Parameters: relationshipCode - specifies the type of relationship; for example, "SUBSTITUTE"

isSQLRule - whether the relationship is defined by a SQL rule

front - beginning index of items to load. Indexing starts at 0.

end - ending index of items to load

Returns: subarray of items related to this item by the relationship code passed in as parameter

getRelatedItems

```
public Item[] getRelatedItems(String relationshipCode, boolean isSQLRule, String
bindArgs[])
throws SQLException, FrameworkException, CatalogException, ItemNotFoundException
```

Retrieves items related to this item by the relationship code passed in as parameter.

Parameters: relationshipCode - specifies the type of relationship; for example, "SUBSTITUTE"

isSQLRule - whether the relationship is defined by a SQL rule

bindArgs - array of Strings containing bind arguments, used in relationships defined using SQL rules with bind arguments

Returns: items related to this item by the relationship code passed in as parameter

getRelatedItems

```
public Item[] getRelatedItems(String relationshipCode, boolean isSQLRule, String
bindArgs[], int front, int end)
throws SQLException, FrameworkException, CatalogException, ItemNotFoundException
```

Retrieves subarray of items related to this item by the relationship code passed in as parameter. The subarray will contain items whose position in the related item ID array is between the front and end indexes passed in as parameter. If the end index is larger than the number of related items, returns the items from the front index to the end of the list of items.

Parameters: relationshipCode - specifies the type of relationship; for example, "SUBSTITUTE" isSQLRule - whether the relationship is defined by a SQL rule

bindArgs - array of Strings containing bind arguments, used in relationships defined using SQL rules with bind arguments

front - beginning index of items to load. Indexing starts at 0.

end - ending index of items to load

Returns: subarray of items related to this item by the relationship code passed in as parameter

getRelatedPrice

```
public PriceObject getRelatedPrice(Item itm)
throws FrameworkException, SQLException, CatalogException
```

Retrieves price of an item whose price is based on this item's price. This API can be used to retrieve the price of items whose price depends on the price of another item. Prices retrieved are based on the minisite's price list ID and the primary UOM codes of the items. If minisite price list ID is null, passes party ID and account ID to pricing engine to determine the price list. Returns PriceObject containing the price of the item passed in as parameter. Call getListPrice(), getBestPrice() on the PriceObject returned to get the price of the item whose price depends on this item.

Parameters: itm - item whose price depends on the price of this base item

Returns: PriceObject - PriceObject containing the price of the item passed in as parameter

getRelatedPrice

```
public PriceObject getRelatedPrice(Item itm, BigDecimal priceListId)
throws FrameworkException, SQLException, CatalogException
```

Retrieves price of an item whose price is based on this item's price. This API can be used to retrieve the price of items whose price depends on the price of another item. Prices retrieved are based on the price list ID passed in and the primary UOM codes of the items. If price list ID is null, passes party ID and account ID to pricing engine to determine the price list. Returns PriceObject containing the price of the item passed in as parameter. Call getListPrice(), getBestPrice() on the PriceObject returned to get the price of the item whose price depends on this item.

Parameters: itm - item whose price depends on the price of this base item

priceListId - price list ID

Returns: PriceObject - PriceObject containing the price of the item passed in as parameter

getRelatedPrice

```
public PriceObject getRelatedPrice(Item itm, BigDecimal partyId, BigDecimal
accountId)
throws FrameworkException, SQLException, CatalogException
```

Retrieves price of an item whose price is based on this item's price. This API can be used to retrieve the price of items whose price depends on the price of another item. Prices retrieved are based on the minisite price list, partyId, accountId, and the primary UOM codes. Returns PriceObject containing the price of the item passed in as parameter. Call getListPrice(), getBestPrice() on the PriceObject returned to get the price of the item whose price depends on this item.

Parameters: itm - item whose price depends on the price of this base item

partyId - customer's party ID

accountId - customer's account ID

Returns: PriceObject - PriceObject containing the price of the item passed in as parameter

getRelatedPrice

```
public PriceObject getRelatedPrice(Item itm, BigDecimal priceListId, BigDecimal
partyId, BigDecimal accountId)
throws FrameworkException, SQLException, CatalogException
```

Retrieves price of an item whose price is based on this item's price. This API can be used to retrieve the price of items whose price depends on the price of another item. Prices retrieved are based on the priceListId, partyId, accountId, and the primary UOM codes. If priceListId is null, partyId and accountId will be used to determine which price list to use for pricing. Otherwise, uses the price list passed in as parameter. Returns PriceObject containing the price of the item passed in as parameter. Call getListPrice(), getBestPrice() on the PriceObject returned to get the price of the item whose price depends on this item.

Parameters: itm - item whose price depends on the price of this base item

priceListId - price list to use for pricing

partyId - customer's party ID

accountId - customer's account ID

Returns: PriceObject - PriceObject containing the price of the item passed in as parameter

getRelatedPrices

```
public PriceObject[] getRelatedPrices(Item itms[])
throws FrameworkException, SQLException, CatalogException
```

Retrieves prices of items whose price is based on this item's price. This API can be used to retrieve the price of items whose price depends on the price of another item. Prices retrieved are based on the minisite's price list ID and the primary UOM codes of the items. If minisite's price list is null, uses party ID and account ID to determine which price list to use. Returns PriceObject[] the size of the Item[] passed in as parameter. The price contained in PriceObject[i] corresponds to the price of Item[i]. To retrieve the price of Item[i], call PriceObject[i].getListPrice() or PriceObject[i].getBestPrice().

Parameters: itms - array of items whose prices depend on the price of this base item

Returns: PriceObject[] - array of PriceObject containing the price of each item in the item array

getRelatedPrices

```
public PriceObject[] getRelatedPrices(Item itms[], BigDecimal priceListId)
throws FrameworkException, SQLException, CatalogException
```

Retrieves prices of items whose price is based on this item's price. This API can be used to retrieve the price of items whose price depends on the price of another item. Prices retrieved are based on the price list ID passed in and the primary UOM codes of the items. If price list ID is null, uses party ID and account ID to determine which price list to use. Returns PriceObject[] the size of the Item[] passed in as parameter. The price contained in PriceObject[i] corresponds to the price of Item[i]. To retrieve the price of Item[i], call PriceObject[i].getListPrice() or PriceObject[i].getBestPrice().

Parameters: itms - array of items whose prices depend on the price of this base item

priceListId - price list ID

Returns: PriceObject[] - array of PriceObject containing the price of each item in the item array

getRelatedPrices

```
public PriceObject[] getRelatedPrices(Item itms[], BigDecimal partyId,
BigDecimal accountId)
```

```
throws FrameworkException, SQLException, CatalogException
```

Retrieves prices of items whose price is based on this item's price. This API can be used to retrieve the price of items whose price depends on the price of another item. Prices retrieved are based on the partyId, accountId, and the primary UOM codes of the items. Returns PriceObject[] the size of the Item[] passed in as parameter. The price contained in PriceObject[i] corresponds to the price of Item[i]. To retrieve the price of Item[i], call PriceObject[i].getListPrice() or PriceObject[i].getBestPrice().

Parameters: itms - array of items whose prices depend on the price of this base item

partyId - customer's party ID

accountId - customer's account ID

Returns: PriceObject[] - array of PriceObject containing the price of each item in the item array

getRelatedPrices

```
public PriceObject[] getRelatedPrices(Item itms[], BigDecimal priceListId,
BigDecimal partyId, BigDecimal accountId)
throws FrameworkException, SQLException, CatalogException
```

Retrieves prices of items whose price is based on this item's price. This API can be used to retrieve the price of items whose price depends on the price of another item. Prices retrieved are based on the priceListId, partyId, accountId, and the primary UOM codes of the items. If priceListId is null, uses the partyId and accountId to determine which price list to use for pricing. Otherwise, uses the price list passed in as parameter. Returns PriceObject[] the size of the Item[] passed in as parameter. The price contained in PriceObject[i] corresponds to the price of Item[i]. To retrieve the price of Item[i], call PriceObject[i].getListPrice() or PriceObject[i].getBestPrice().

Parameters: itms - array of items whose prices depend on the price of this base item

priceListId - price list used for pricing

partyId - customer's party ID

accountId - customer's account ID

Returns: PriceObject[] - array of PriceObject containing the price of each item in the item array

getRelatedSectionIDs

```
public int[] getRelatedSectionIDs(String relationshipCode)
```

```
throws SQLException, FrameworkException, SectionNotFoundException
```

Retrieves IDs of sections related to this item by the relationship code passed in as pararmeter

Parameters: relationshipCode - specifies the type of relationship; for example "SUBSTITUTE"

Returns: IDs of sections related to this item by the relationship code passed in as parameter

getSegmentColumn

```
public String getSegmentColumn(int k)
throws InvalidColumnNumberException, FrameworkException, SQLException,
ItemNotFoundException
```

Retrieves the value of column from Segment1-20 in MTL_SYSTEM_ITEMS_VL

Parameters: k - int representing which segment to return

Returns: segment value in column MTL_SYSTEM_ITEMS.SEGMENTk

getSrvcDuration

```
public int getSrvcDuration()
throws SQLException, FrameworkException, ItemNotFoundException
```

Retrieves default service duration

Returns: default service duration

getSrvcPeriod

```
public String getSrvcPeriod()
throws SQLException, FrameworkException, ItemNotFoundException
```

Retrieves item’s period for default service duration

Returns: period for default service duration

getSrvcStartDelay

```
public int getSrvcStartDelay()
throws SQLException, FrameworkException, ItemNotFoundException
```

Retrieves number of days after shipment that service begins

Returns: days after shipment that service begins

getTemplateFileName

```
public String getTemplateFileName(int dispCtxID)
throws FrameworkException, SQLException
```

Retrieves the file name of the physical template associated with this item for a particular display context.

Parameters: dispCtx - display context ID for the template

Returns: file name of physical template. If the required template is not found, returns null.

getTemplateFileName

```
public String getTemplateFileName(String displayContext)
throws FrameworkException, SQLException
```

Retrieves the file name of the physical template associated with this item for a particular display context.

Parameters: displayContext - display context for the template; for example, STORE_PRODUCT_DETAILS, STORE_PRODUCT_DESCR

Returns: file name of the physical template. If the required template is not found, returns null.

getUOM

```
public String getUOM(String uomCode)
throws SQLException, FrameworkException, ItemNotFoundException
```

Retrieves translated UOM based on the user’s language for the UOM code passed in as parameter. Returns null if the UOM code is not in the item’s list of UOM codes.

Parameters: uomCode - UOM codes used to get the translated UOM

Returns: translated UOM based on the user’s language

getUOMCodes

```
public String[] getUOMCodes()
```

If the profile option IBE: Retrieve All Units of Measure for an Item is set to ‘Yes’ or does not have a value, retrieves all the UOM codes defined for the item. If the profile option IBE: Retrieve All Units of Measure for an Item is set to ‘No’, retrieves the primary UOM code.

Returns: item’s UOM codes

getUOMs

```
public String[] getUOMs()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves units of measure, based on the user's language

Returns: String[ ] containing item's UOMs based on the user's language. String[ ] of size 0 if no units of measures were found. For UOMs where a translated version cannot be found for the user's language, an empty string will be returned.

isBackOrderable

```
public boolean isBackOrderable()
```

Retrieves whether item can be back ordered

Returns: true if item can be back ordered, false otherwise

isBomEnabled

```
public boolean isBomEnabled()
```

Retrieves whether item is BOM enabled

Returns: true if item is BOM enabled, false otherwise

isConfigurable

```
public boolean isConfigurable()
```

Retrieves whether item can be configured

Returns: true if item can be configured, false otherwise

isCouponExempt

```
public boolean isCouponExempt()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves whether item is coupon exempt

Returns: true if item is coupon exempt, false otherwise

isDownloadable

```
public boolean isDownloadable()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves whether item is downloadable

Returns: true if item is downloadable, false otherwise

isElectronic

```
public boolean isElectronic()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves whether item is electronic

Returns: true if item is electronic, false otherwise

isOrderable

```
public boolean isOrderable()
```

Retrieves whether item is orderable via Web

Returns: true if item is orderable on the Web, false otherwise

isReturnable

```
public boolean isReturnable()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves whether item is returnable

Returns: true if item is returnable, false otherwise

isService

```
public boolean isService()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves whether item is a service item

Returns: true if item is a service item, false otherwise

isServiceable

```
public boolean isServiceable()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves whether item is serviceable

Returns: true if item is serviceable, false otherwise

isShippable

```
public boolean isShippable()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves whether item is shippable

Returns: true if item is shippable, false otherwise

isTaxable

```
public boolean isTaxable()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves whether item is taxable

Returns: true if item is taxable, false otherwise

isVolDiscountExempt

```
public boolean isVolDiscountExempt()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves whether item is volume discount exempt

Returns: true if item is volume discount exempt, false otherwise

load

```
public static Item load(int itemID)
throws SQLException, FrameworkException, ItemNotFoundException
```

Loads the item with the inventory_item_id passed in as parameter. Load level of the item will be SHALLOW. If the profile option IBE: Retrieve Price When Displaying Items is set to 'Yes', loads the price for the item's primary UOM. Otherwise, does not load the price.

Parameters: itemID - item ID corresponding to MTL_SYSTEM_ITEMS_
VL.INVENTORY_ITEM_ID

Returns: item with values loaded for the proper members

load

```
public static Item load(int itemID, int mode)
throws SQLException, FrameworkException, ItemNotFoundException
```

Loads the item with the inventory_item_id passed in as parameter. If the profile option IBE: Retrieve Price When Displaying Items is set to 'Yes', loads the price for the item's primary UOM. Otherwise, does not load the price.

Parameters: itemID - item ID corresponding to MTL_SYSTEM_ITEMS_ VL.INVENTORY_ITEM_ID

mode - load level for the item; Item.SHALLOW or Item.DEEP

Returns: item with values loaded for the proper members

load

```
public static Item load(int itemID, int mode, boolean retrievePrice)
throws SQLException, FrameworkException, ItemNotFoundException
```

Loads the item with the inventory_item_id and mode passed in as parameter. Uses the retrievePrice parameter to determine whether to retrieve the price for the primary UOM of the item.

Parameters: itemID - item ID corresponding to MTL_SYSTEM_ITEMS_ VL.INVENTORY_ITEM_ID

mode - load level for the item; Item.SHALLOW or Item.DEEP

retrievePrice - whether to retrieve the price for the item's primary UOM

Returns: item with values loaded for the proper members

load

```
public static Item load(String partNum)
throws SQLException,  FrameworkException, ItemNotFoundException,
CatalogException
```

Loads the item with the part number passed in as parameter. Load level of the item will be SHALLOW. If the profile option IBE: Retrieve Price When Displaying Items is set to 'Yes', loads the price for the item's primary UOM. Otherwise, does not load the price.

Parameters: partNum - item part number corresponding to MTL_SYSTEM_ITEMS_ VL.CONCATENATED_SEGMENTS

Returns: item with values loaded for the proper members

load

```
public static Item load(String partNum, int mode)
throws SQLException, FrameworkException, ItemNotFoundException, CatalogException
```

Loads the item with the part number and mode passed in as parameter. If the profile option IBE: Retrieve Price When Displaying Items is set to 'Yes', loads the price for the item's primary UOM. Otherwise, does not load the price.

Parameters: partNum - item part number corresponding to MTL_SYSTEM_ITEMS_VL.CONCATENATED_SEGMENTS

mode - load level for the item; Item.SHALLOW or Item.DEEP

Returns: item with values loaded for the proper members

load

```
public static Item load(String partNum, int mode, boolean retrievePrice)
throws SQLException, FrameworkException, CatalogException, ItemNotFoundException
```

Loads the item with the part number and mode passed in as parameter. Uses the retrievePrice parameter to determine whether to retrieve the price for the primary UOM of the item.

Parameters: partNum - item part number corresponding to MTL_SYSTEM_ITEMS_VL.CONCATENATED_SEGMENTS

mode - load level for the item; Item.SHALLOW or Item.DEEP

retrievePrice - whether to retrieve the price for the item's primary UOM

Returns: item with values loaded for the proper members

load

```
public static Item[] load(int itemIDs[])
throws SQLException, FrameworkException, CatalogException, ItemNotFoundException
```

Loads the items with the item IDs passed in as parameter. The load level of the items will be SHALLOW. If the profile option IBE: Retrieve Price When Displaying Items is set to 'Yes', loads the price for each item's primary UOM. Otherwise, does not load the price. The order of the items will be in the order of the IDs passed in. Duplicates and items not found in the database will be removed from the Item array.

Parameters: itemIDs - item IDs corresponding to MTL_SYSTEM_ITEMS_
VL.INVENTORY_ITEM_ID

Returns: array of items with values loaded for the proper members

load

```
public static Item[] load(int itemIDs[], int mode)
throws SQLException, FrameworkException, CatalogException, ItemNotFoundException
```

Loads items with the array of item IDs and mode passed in as parameter. If the profile option IBE: Retrieve Price When Displaying Items is set to 'Yes', loads the price for each item's primary UOM. Otherwise, does not load the price. The order of the items will be in the order of the IDs passed in. Duplicates and items not found in the database will be removed from the Item array.

Parameters: itemIDs - array of item IDs corresponding to mtl_sysetm_items_
vl.inventory_item_id

mode - load level for the item; Item.SHALLOW or Item.DEEP

Returns: array of items with values loaded for the proper members

load

```
public static Item[] load(int itemIDs[], int mode, boolean retrievePrice)
throws SQLException, FrameworkException, CatalogException, ItemNotFoundException
```

Load the items with the item IDs and mode passed in as parameter. Uses the retrievePrice parameter to determine whether to retrieve the price for each item's primary UOM. The order of the items will be in the order of the IDs passed in. Duplicates and items not found in the database will be removed from the Item array.

Parameters: itemIDs - array of item IDs corresponding to MTL_SYSTEM_ITEMS_
VL.INVENTORY_ITEM_ID

mode - load level for the item; Item.SHALLOW or Item.DEEP

retrievePrice - whether to retrieve the prices for the item's primary UOM

Returns: array of items with values loaded for the proper members

load

```
public static Item[] load(int itemIDs[], int mode, boolean retrievePrice,
boolean compact)
throws SQLException, FrameworkException, CatalogException, ItemNotFoundException
```

Loads the items with the item IDs and mode passed in as parameter. Uses the retrievePrice parameter to determine whether to retrieve the price for each item's primary UOM. The order of the items will be in the order of the IDs passed in. If compact is true, removes items that were not found in the database. If compact is false, returns a null for items not found in the database.

Parameters: itemIDs - array of item IDs corresponding to MTL_SYSTEM_ITEMS_VL,INVENTORY_ITEM_ID

mode - load level for the item; Item.SHALLOW or Item.DEEP

retrievePrice - whether to retrieve the price for the item's primary UOM

compact - whether to remove items not found in the database

Returns: array of items with values loaded for the proper members

load

```
public static Item[] load(String partNums[])
throws SQLException, FrameworkException, ItemNotFoundException, CatalogException
```

Loads the items with the part numbers passed in as parameter. The load level of the items will be SHALLOW. If the profile option IBE: Retrieve Price When Displaying Items is set to 'Yes', loads the price for each item's primary UOM. Otherwise, does not load the price. The order of the items will be in the order of the part numbers passed in. Duplicates and items not found in the database will be removed from the Item array.

Parameters: partNums - array of item part numbers corresponding to MTL_SYSTEM_ITEMS_VL.CONCATENATED_SEGMENTS

Returns: array of items with values loaded for the proper members

load

```
public static Item[] load(String partNums[], int mode)
throws SQLException, FrameworkException, CatalogException, ItemNotFoundException
```

Loads the items with the part numbers and mode passed in as parameter. If the profile option IBE: Retrieve Price When Displaying Items is set to 'Yes', loads the price for each item's primary UOM. Otherwise, does not load the price. The order of the items will be in the order of the part numbers passed in. Duplicates and items not found in the database will be removed from the Item array.

Parameters: partNums - array of item part numbers corresponding to mtl_system_item_vl.concatenated_segments

mode - load level for the items; Item.SHALLOW or Item.DEEP

Returns: array of items with values loaded for the proper members

load

```
public static Item[] load(String partNums[], int mode, boolean retrievePrice)
throws SQLException, FrameworkException, CatalogException, ItemNotFoundException
```

Loads the items with the part numbers and mode passed in as parameter. Uses the parameter retrievePrice to determine whether to retrieve the price for each item's primary UOM. The order of the items will be in the order of the part numbers passed in. Duplicates and items not found in the database will be removed from the Item array.

Parameters: partNums - array of item part numbers corresponding to MTL_SYSTEM_ITEMS_VL.CONCATENATED_SEGMENTS

mode - load level for the items; Item.SHALLOW or Item.DEEP

retrievePrice - whether to retrieve the price for each item's primary UOM

Returns: array of items with values loaded for the proper members

validateQuantity

```
public static boolean validateQuantity(int itemId, BigDecimal organizationId,
String inputQty, String uomCode)
throws FrameworkException, SQLException, ItemNotFoundException
```

Determines whether the input quantity is valid for the item. If the item is OM: Indivisble or serial code controlled, the quantity must be an integer when converted to the primary UOM. Otherwise, the quantity can be a decimal value.

Parameters: itemId - item ID corresponding to MTL_SYSTEM_ITEMS_VL.INVENTORY_ITEM_ID

organizationId - inventory organization ID corresponding to MTL_SYSTEM_ITEMS_VL.ORGANIZATION_ID

inputQty - input quantity to validate

uomCode - UOM code

Returns: true if the input quantity is valid for the item and UOM, false otherwise

validateQuantity

```
public static boolean[] validateQuantity(int itemIds[], BigDecimal
organizationId[], String inputQty[], String uomCodes[])
throws FrameworkException, SQLException, ItemNotFoundException
```

For each item, determines whether the input quantity is valid. If an item is OM: Indivisble or serial code controlled, the quantity must be an integer when converted to the primary UOM. Otherwise, the quantity can be a decimal value.

Parameters: itemIds - item IDs corresponding to MTL_SYSTEM_ITEMS_ VL.INVENTORY_ITEM_ID

organizationId - inventory organization IDs corresponding to MTL_SYSTEM_ ITEMS_VL.ORGANIZATION_ID

inputQty - input quantity to validate

uomCodes - UOM code

Returns: boolean array indicating whether each input quantity is valid for the given item and UOM. If inputQty[i] is valid for itemIds[i], organizationId[i], and uomCodes[i], the value of boolean[i] is true. Otherwise, the valie of boolean[i] is false.

3.3 Class ItemFlexfield

```
java.lang.Object > oracle.apps.ibe.catalog.ItemFlexfield
```

public class **ItemFlexfield**

extends Object

The ItemFlexfield object contains the segment information for an item flexfield segment. It is used to retrieve the name, prompt, value, and database column name for an item flexfield segment. This object is returned by the getFlexfields() methods in the Item class.

3.3.1 Methods for Class ItemFlexfield

The following table is an index of Class ItemFlexfield methods:

Table 3–3 Method Index for Class ItemFlexfield

Method	Description
getDbColumnName	Retrieves the database column name of the flexfield segment

Table 3–3 Method Index for Class ItemFlexfield (Cont.)

Method	Description
getName	Retrieves the name of the flexfield segment
getPrompt	Retrieves the prompt of the flexfield segment
getValue	Retrieves the value of the flexfield segment

getDbColumnName

```
public String getDbColumnName()
```

Retrieves the database column name of the flexfield segment

Returns: database column name of the flexfield segment

getName

```
public String getName()
```

Retrieves the name of the flexfield segment

Returns: name of the flexfield segment

getPrompt

```
public String getPrompt()
```

Retrieves the prompt of the flexfield segment

Returns: prompt of the flexfield segment

getValue

```
public String getValue()
```

Retrieves the value of the flexfield segment

Returns: value of the flexfield segment

3.4 Class PriceObject

```
java.lang.Object > oracle.apps.ibe.catalog.PriceObject
```

public class **PriceObject**

extends Object

The PriceObject contains pricing information retrieved for an item. It is used to retrieve list price and selling price. It also provides the functionality to format a price based on currency. This object is returned by the getListAndBestPrices() APIs in the Item class.

3.4.1 Methods for Class PriceObject

The following table is an index of Class PriceObject methods:

Table 3–4 Method Index for Class PriceObject

Method	Description
formatNumber	Formats the price based on currency code
getBestPrice	Retrieves the best price stored in the PriceObject
getListPrice	Retrieves the list price stored in the PriceObject

formatNumber

```
public static String formatNumber(String currencyCode, BigDecimal number)
throws FrameworkException, SQLException
```

Formats the price based on currency code. Prepends currency symbol to the front of the number, adds the appropriate decimal and grouping separators. If currencyCode is null, uses Java's default formatting. If number is null, returns an empty string.

Parameters: currencyCode - currency code used to format the price

number - price to be formatted

Returns: string containing price formatted based on currency code

formatNumber

```
public static String formatNumber(String currencyCode, double number)
throws FrameworkException, SQLException
```

Format the price based on currency code. Prepends currency symbol to the front of the number, adds the appropriate decimal and grouping separators. If currencyCode is null, uses Java's default formatting.

Parameters: currencyCode - currency code used to format the price number - price to be formatted

Returns: string containing price formatted based on currency code

getBestPrice

```
public BigDecimal getBestPrice()
throws PriceNotFoundException
```

Retrieves the best price stored in the PriceObject

Returns: best price store in the PriceObject

getListPrice

```
public BigDecimal getListPrice()
throws PriceNotFoundException
```

Retrieves the list price stored in the PriceObject

Returns: list price stored in the PriceObject

3.5 Class Section

```
java.lang.Object > oracle.apps.ibe.catalog.Section
```

public class **Section**

extends Object

The Section object is the building block of the display hierarchy. There are two types of sections: FEATURED and NAVIGATIONAL. Featured sections cannot contain subsections and are not displayed in the browse hierarchy of the store. Navigational sections can contain subsections and are displayed in the browse hierarchy of the store. Each section (except the hierarchy root) has one parent section and one or more subsections. A section with subsections is a non-leaf section. A section without subsections is a leaf section. Leaf sections are the only sections that can contain items.

The Section object is used to retrieve basic section attributes stored in jtf_dsp_sections_vl (such as display name, description, long description, etc.). It is also used to retrieve the template associated to the section, the media associated to the section for a specific display context, the list of supersections, the list of subsections of a certain type (FEATURED or NAVIGATIONAL), the list of items, the list of sibling sections, and the list of related sections.

3.5.1 Variables for Class Section

SHALLOW

```
public static final int SHALLOW
```

SHALLOW is a constant passed into Section load APIs to request a Section shallow load, which will load the following Section attributes SECTION_ID, ACCESS_NAME, DISPLAY_NAME, DESCRIPTION, OBJECT_VERSION_NUMBER, SECTION_TYPE_CODE. If information other than the above is requested from a shallow loaded section, the section will automatically be DEEP loaded, after which all the information will be available.

DEEP

```
public static final int DEEP
```

DEEP is a constant passed into Section load APIs to request a Section deep load, which will load all section attributes, subsections IDs, item IDs, supersection IDs.

NAVIGATIONAL

```
public static final String NAVIGATIONAL
```

Constant for Navigational section type: those sections that can be browsed through navigation. Out of the box, iStore has NAVIGATIONAL and FEATURED section types. The merchant can define new section types.

FEATURED

```
public static final String FEATURED
```

Constant for Featured section type: those sections that are not browsed through navigation.

3.5.2 Methods for Class Section

The following table is an index of Class Section methods:

Table 3–5 Method Index for Class Section

Method	Description
getAccessName	Retrieves section access name
getAttributeCategory	Retrieves section's attribute_category column

Table 3–5 Method Index for Class Section (Cont.)

Method	Description
getAttributeColumn	Retrieves an attribute column value of the section
getDescription	Retrieves description of the section in the current language
getDisplayContextID	Retrieves the display context for items in the section. This method is only applicable to leaf sections.
getDisplayName	Retrieves display name of the section in the current language
getFeaturedItemIDs	Retrieves an array containing the IDs of the items in the section’s featured subsections
getFeaturedItems	Retrieves an array containing the items in the section’s featured subsections
getFeaturedSubSectionIDs	Retrieves an array of featured subsection IDs. If the profile option IBE: Use Catalog exclusions is set to ‘Yes’, minisite exclusions will be removed.
getFeaturedSubSections	Retrieves an array of featured subsections
getItemIDs	Retrieves array of item IDs in current section for the user's validation organization ID. If the profile option IBE: Use Catalog exclusions is set to ‘Yes’, minisite exclusions will be removed.
getItems	Retrieves an array of items in current section. If the profile option IBE: Use Catalog exclusions is set to ‘Yes’, minisite exclusions will be removed.
getLeafSubSectionIDs	Retrieves an array of leaf level descendent subsection IDs. If the profile option IBE: Use Catalog exclusions is set to ‘Yes’, minisite exclusions will be removed.
getLeafSubSections	Retrieves an array of leaf level descendent subsections
getLongDescription	Retrieves long description of the section in the current language
getMediaFileName	Retrieves the file name of the physical media associated with the section for a particular display context
getNonNavSubSectionIDs	Retrieves an array of all non-navigational subsection IDs
getNonNavSubSections	Retrieves an array of non-navigational subsections
getNumberOfItems	Retrieves the number of item IDs within the section available in the user’s validation organization. If the profile option IBE: Use Catalog exclusions is set to ‘Yes’, minisite exclusions will be removed.

Table 3–5 Method Index for Class Section (Cont.)

Method	Description
getRelatedSectionIDs	Retrieves an array of section IDs for all sections related to this section by the relationship code passed in as parameter
getRelatedSections	Retrieves an array of sections related to this section by the relationship code passed in as parameter
getSectionID	Retrieves section ID
getSectionType	Retrieves section type
getSiblingSectionIDs	Retrieves an array of section IDs containing the IDs of the same level siblings
getSiblingSections	Retrieves an array of sections containing the same level siblings
getSubSectionIDs	Retrieves an array of navigational subsection IDs
getSubSectionItemIDs	Retrieves an array containing the IDs of items in subsections with the specified section type
getSubSectionItems	Retrieves an array containing items in a subsection with the specified section type
getSubSections	Retrieves an array of navigational subsections
getSuperSection	Retrieves the immediate super section
getSuperSectionID	Retrieves ID of the immediate super section
getSuperSectionIDs	Retrieves an array of supersection IDs, starting with the minisite root section
getSuperSections	Retrieves an array of supersections, starting with the minisite root section
getTemplateFileName	Retrieves the file name of the physical template associated with the section
isFeatured	Determines whether the section is a featured section
isLeafSection	Determines whether this section is a leaf section
load	Loads a section with the parameters passed in

getAccessName

```
public String getAccessName()
```

Retrieves section access name

Returns: access name of the section

getAttributeCategory

```
public String getAttributeCategory()
throws SQLException, FrameworkException, SectionNotFoundException
```

Retrieves section's attribute_category column

Returns: section's attribute category

getAttributeColumn

```
public String getAttributeColumn(int k)
throws FrameworkException, SQLException, SectionNotFoundException,
InvalidColumnNumberException
```

Retrieves an attribute column value of the section

Parameters: k - the attribute column number, 1 to 15

Returns: attribute value in column JTF_DSP_SECTIONS_VL.ATTRIBUTEk

getDescription

```
public String getDescription()
throws SQLException, FrameworkException, SectionNotFoundException
```

Retrieves description of the section in the current language

Returns: section description

getDisplayContextID

```
public int getDisplayContextID()
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves the display context for items in the section. This method is only applicable to leaf sections.

Returns: display context ID defined for this section. If jtf_dsp_sections_vl.display_ context_id is null, returns -1.

getDisplayName

```
public String getDisplayName()
throws SQLException, FrameworkException, SectionNotFoundException
```

Retrieves display name of the section in the current language

Returns: section display name

getFeaturedItemIDs

```
public int[] getFeaturedItemIDs()
throws FrameworkException, SQLException, SectionNotFoundException,
ItemNotFoundException
```

Retrieves an array containing the IDs of the items in the section's featured subsections.

Returns: array of item IDs

getFeaturedItems

```
public Item[] getFeaturedItems()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves an array containing the items in the section's featured subsections.

Returns: array of items

getFeaturedSubSectionIDs

```
public int[] getFeaturedSubSectionIDs()
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves an array of featured subsection IDs. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Returns: array of featured subsection IDs

getFeaturedSubSections

```
public Section[] getFeaturedSubSections()
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves an array of featured subsections. The load level of the sections will be SHALLOW. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Returns: array of featured subsections

getItemIDs

```
public int[] getItemIDs()
```

```
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves array of item IDs in current section for the user's validation organization ID. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Returns: array of item IDs

getItemIDs

```
public int[] getItemIDs(String ordByClause)
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves an array of item IDs ordered using the ordByClause parameter.

Parameters: orderByClause - comma-separated list of columns in MTL_SYSTEM_ITEMS_VL that will be used to order the items; for example "description asc, inventory_item_id, concatenated_segments desc"

Returns: array of item IDs ordered by the ordByClause passed in as parameter

getItems

```
public Item[] getItems()
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves an array of items in current section. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Returns: array of items

getItems

```
public Item[] getItems(int front)
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves a subarray of items in current section. The subarray will start with the item whose position in the item ID array is the front index passed in as parameter. It will end after retrieving number of items shown on a page (profile option IBE: Items Per Page for Display).

Parameters: front - index indicating the position in the item ID array at which to start retrieving items; indexing starts at 0

Returns: subarray of items

getItems

```
public Item[] getItems(int front, int end)
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves a subarray of items in current section. The subarray will contain items whose position in the item ID array is between the front and end indexes passed in as parameter.

Parameters: front - index indicating the position in the item ID array at which to start retrieving items; indexing starts at 0

end - index indicating the position in the item ID array at which to stop retrieving items

Returns: subarray of items

getItems

```
public Item[] getItems(String orderByClause)
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves an array of items in current section, ordered by the orderByClause parameter. This method will always go to the database to retrieve the item IDs.

Parameters: orderByClause - comma-separated list of columns in MTL_SYSTEM_ITEMS_VL used to order the items; for example, "description asc, inventory_item_id, concatenated_segments desc"

Returns: array of items, ordered by the orderByClause parameter

getItems

```
public Item[] getItems(String orderByClause, int front)
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves a subarray of items, ordered by the orderByClause parameter. The subarray will start with the item whose position in the item ID array is the front index passed in as parameter. It will end after retrieving number of items shown on a page (profile option IBE: Items Per Page for Display). This method will always go to the database to retrieve the item IDs.

Parameters: orderByClause - comma-separated list of columns in MTL_SYSTEM_ITEMS_VL used to order the items; for example, "description asc, inventory_item_id, concatenated_segments desc" front - index indicating the position at which to start retrieving items; indexing starts at 0

Returns: subarray of items, ordered by the orderByClause parameter

getItems

```
public Item[] getItems(String orderByClause, int front, int end)
throws FrameworkException, SQLException, ItemNotFoundException
```

Retrieves a subarray of items, ordered by the orderByClause parameter. The subarray will contain items whose position in the array is between the front and end indexes passed in as parameter. This method will always go to the database to retrieve the item IDs.

Parameters: orderByClause - comma-separated list of columns in MTL_SYSTEM_ITEMS_VL used to order the items; for example, "description asc, inventory_item_id, concatenated_segments desc"

front - index indicating the position at which to start retrieving items; indexing starts at 0

end - index indicating the position in the item ID array at which to stop retrieving items

Returns: subarray of items, ordered by the orderByClause parameter

getLeafSubSectionIDs

```
public int[] getLeafSubSectionIDs()
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves an array of leaf level descendent subsection IDs. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Returns: array of leaf level descendent subsection IDs

getLeafSubSections

```
public Section[] getLeafSubSections()
throws FrameworkException, SQLException, SectionNotFoundException,
CatalogException
```

Retrieves an array of leaf level descendent subsections. The load level of the sections will be SHALLOW. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Returns: array of leaf level descendent subsections

getLongDescription

```
public String getLongDescription()
throws SQLException, FrameworkException, SectionNotFoundException
```

Retrieves long description of the section in the current language

Returns: section long description

getMediaFileName

```
public String getMediaFileName(String mediaCtx)
throws FrameworkException, SQLException
```

Retrieves the file name of the physical media associated with the section for a particular display context

Parameters: mediaCtx - display context for the media file; for example, STORE_SECTION_SMALL_IMAGE

Returns: File name of the media. If the required media is not found, returns null.

getNonNavSubSectionIDs

```
public int[] getNonNavSubSectionIDs()
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves an array of all non-navigational subsection IDs. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Returns: array of non-navigational subsection IDs

getNonNavSubSections

```
public Section[] getNonNavSubSections()
throws SQLException, FrameworkException, SectionNotFoundException
```

Retrieves an array of non-navigational subsections. The load level of the sections will be SHALLOW. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Returns: array of non-navigational subsections

getNumberOfItems

```
public int getNumberOfItems()
```

```
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves the number of item IDs within the section available in the user's validation organization. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Returns: number of item IDs in the section available in the user's organization ID.

getRelatedSectionIDs

```
public int[] getRelatedSectionIDs(String relationCode)
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves an array of section IDs for all sections related to this section by the relationship code passed in as parameter. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Parameters: String - relationCode - specifies the type of relationship; for example, "SUBSTITUTE"

Returns: array of related section IDs

getRelatedSections

```
public Section[] getRelatedSections(String relationCode)
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves an array of sections related to this section by the relationship code passed in as parameter. The load level of the related sections will be SHALLOW. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Returns: array of related sections

getSectionID

```
public int getSectionID()
```

Retrieves section ID

Returns: the ID of the section

getSectionType

```
public String getSectionType()
```

Retrieves section type

Returns: section type of the section

getSiblingSectionIDs

```
public int[] getSiblingSectionIDs(boolean includeSelf)
throws SQLException, FrameworkException, SectionNotFoundException
```

Retrieves an array of section IDs containing the IDs of the same level siblings. Uses the includeSelf parameter to determine whether to return this section's ID in the array. This section's ID should be returned when it is necessary to preserve the ordering of this section within its list of siblings. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Parameters: includeSelf - whether to return this section's ID in the array

Returns: array of section IDs

getSiblingSections

```
public Section[] getSiblingSections(boolean includeSelf)
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves an array of sections containing the same level siblings. The load level of the sections will be SHALLOW. Uses the includeSelf parameter to determine whether to return this section in the array. This section should be returned when it is necessary to preserve the ordering of this section within its list of siblings. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Parameters: includeSelf - whether to return this section in the array

Returns: array of sections

getSubSectionIDs

```
public int[] getSubSectionIDs()
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves an array of navigational subsection IDs. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Returns: array of navigational subsection IDs

getSubSectionIDs

```
public int[] getSubSectionIDs(String sectType)
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves an array of subsection IDs with the section type passed in as parameter. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Parameters: sectType - section type; Section.NAVIGATIONAL, Section.FEATURED

Returns: array of subsection IDs for those subsections with the specified section type

getSubSectionItemIDs

```
public int[] getSubSectionItemIDs(String sectionType)
throws FrameworkException, SQLException, ItemNotFoundException,
SectionNotFoundException
```

Retrieves an array containing the IDs of items in subsections with the specified section type. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Parameters: sectionType - section type; Section.NAVIGATIONAL or Section.FEATURED

Returns: array of item IDs

getSubSectionItems

```
public Item[] getSubSectionItems(String sectionType)
throws FrameworkException, SQLException, ItemNotFoundException,
SectionNotFoundException
```

Retrieves an array containing items in a subsection with the specified section type. The load level of the items will be SHALLOW. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Parameters: sectionType - section type; Section.NAVIGATIONAL or Section.FEATURED

Returns: array of items

getSubSections

```
public Section[] getSubSections()
throws SQLException, FrameworkException, SectionNotFoundException
```

Retrieves an array of navigational subsections. The load level of the sections will be SHALLOW. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Returns: array of navigational subsections

getSubSections

```
public Section[] getSubSections(int num)
throws SQLException, FrameworkException, SectionNotFoundException
```

Retrieves an array of navigational subsections up to the maximum number passed in as parameter. The load level of the sections will be SHALLOW. If the profile option IBE: Use Catalog exclusions is set to Yes, minisite exclusions will be removed.

Parameters: num - maximum number of navigational subsections to return

Returns: array of navigational subsections

getSubSections

```
public Section[] getSubSections(String sectType)
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves an array of subsections with the section type passed in as parameter. The load level of the sections will be SHALLOW. If the profile option IBE: Use Catalog exclusions is set to Yes, minisite exclusions will be removed.

Parameters: sectType - section type; Section.NAVIGATIONAL or Section.FEATURED

Returns: array of subsections for those subsections with the specified section type.

getSuperSection

```
public Section getSuperSection()
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves the immediate super section. The load level of the supersection will be SHALLOW. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Returns: parent section

getSuperSectionID

```
public int getSuperSectionID()
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves ID of the immediate super section. If the profile option IBE: Use Catalog exclusions is set to Yes, minisite exclusions will be removed.

Returns: parent section ID

getSuperSectionIDs

```
public int[] getSuperSectionIDs()
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves an array of supersection IDs, starting with the minisite root section. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Returns: array of supersection IDs, starting with the minisite root section

getSuperSections

```
public Section[] getSuperSections()
throws FrameworkException, SQLException, SectionNotFoundException
```

Retrieves an array of supersections, starting with the minisite root section. If the profile option IBE: Use Catalog exclusions is set to 'Yes', minisite exclusions will be removed.

Returns: array of supersections, starting with the minisite root section

getTemplateFileName

```
public String getTemplateFileName()
throws TemplateNotFoundException, FrameworkException, SQLException
```

Retrieves the file name of the physical template associated with the section

Returns: file name of the template associated with this section

isFeatured

```
public boolean isFeatured()
```

Determines whether the section is a featured section

Returns: true if the section is featured, false otherwise

isLeafSection

```
public boolean isLeafSection()
throws FrameworkException, SQLException, SectionNotFoundException
```

Determines whether this section is a leaf section. A leaf section is a section with no subsections.

Returns: true if the section is a leaf section, otherwise false

load

```
public static Section load(int sectID)
throws FrameworkException, SQLException, SectionNotFoundException
```

Loads a section with the section ID passed in as parameter. The load level of the section will be SHALLOW.

Parameters: sectID - Section ID corresponding to jtf_dsp_sections_vl.section_id

Returns: section with values loaded for the proper members.

load

```
public static Section load(int sectID, int mode)
throws FrameworkException, SQLException, SectionNotFoundException
```

Loads a section with the section ID and mode passed in as parameter.

Parameters: sectID - Section ID corresponding to jtf_dsp_sections_vl.section_id

mode - load level for the section; Section.SHALLOW or Section.DEEP

Returns: section with values loaded for the proper members.

load

```
public static Section load(String accessName)
throws FrameworkException, SQLException, SectionNotFoundException
```

Loads a section with the access name passed in as parameter. The load level of the section will be SHALLOW.

Parameters: accessName - Section access name corresponding to jtf_dsp_sections_vl.access_name

Returns: section with values loaded for the proper members.

load

```
public static Section load(String accessName, int mode)
throws FrameworkException, SQLException, SectionNotFoundException
```

Loads a section with the access name and mode passed in as parameter.

Parameters: accessName - Section access name corresponding to jtf_dsp_sections_vl.access_name

mode - load level for the section; Section.SHALLOW or Section.DEEP

Returns: section with values loaded for the proper members

load

```
public static Section[] load(int sectIDs[])
throws FrameworkException, SQLException, SectionNotFoundException
```

Load the sections with the section IDs passed in as parameter.

Parameters: sectIDs - array of section IDs corresponding to jtf_dsp_sections_vl.section_id

Returns: array of sections with values loaded for the proper members

load

```
public static Section[] load(int sectIDs[], int mode)
throws FrameworkException, SQLException, SectionNotFoundException,
CatalogException
```

Loads the sections with the sectionIDs and mode passed in as parameter.

Parameters: sectIDs - array of section IDs corresponding to jtf_dsp_sections_vl.section_id

mode - load level for the sections; Section.SHALLOW or Section.DEEP

Returns: array of sections with values loaded for the proper members

load

```
public static Section[] load(String accessNames[])
throws SQLException, FrameworkException, SectionNotFoundException,
CatalogException
```

Loads the sections with the access names passed in as parameter. The load level of the sections will be SHALLOW.

Parameters: accessNames - array of section access names corresponding to jtf_dsp_sections_vl.access_name

Returns: array of sections with values loaded for the proper members

load

```
public static Section[] load(String accessNames[], int mode)
throws SQLException, FrameworkException, SectionNotFoundException
```

Loads the sections with the access names and mode passed in as parameter.

Parameters: accessNames - array of section access names corresponding to jtf_dsp_sections_vl.access_name

mode - load level for the section; Section.SHALLOW or Section.DEEP

Returns: array of sections with values loaded for the proper members

3.6 Exception Classes

The methods in the package oracle.apps.ibe.catalog throw the following exceptions:

- SQLException
- FrameworkException
- ItemNotFoundException
- InvalidColumnNumberException
- PriceNotFoundException
- CatalogException
- SectionNotFoundException

3.6.1 SQLException

Exception class to throw if database error occurs.

3.6.2 FrameworkException

Exception class to throw if error occurs while trying to get connection.

3.6.3 ItemNotFoundException

```
java.lang.Object > java.lang.Throwable > java.lang.Exception >
oracle.apps.jtf.base.resources.FrameworkException >
oracle.apps.ibe.catalog.ItemNotFoundException
```

public class **ItemNotFoundException**

extends oracle.apps.jtf.base.resources.FrameworkException

Exception class to throw for Item class methods when item is not found in the database.

Exception class to throw for Section class methods when a section/subsection does not contain any items.

Used by getItems(), getItemIDs(), getSubSectionItemIDs(), getSubSectionItems() in the Section class.

3.6.4 InvalidColumnNumberException

```
java.lang.Object > java.lang.Throwable > java.lang.Exception >
oracle.apps.jtf.base.resources.FrameworkException >
oracle.apps.ibe.catalog.InvalidColumnNumberException
```

public class **InvalidColumnNumberException**

extends oracle.apps.jtf.base.resources.FrameworkException

Exception class to throw when a table attribute column number passed in is not between 1 and 15, or a table segment column number passed in is not between 1 and 20.

Used by getAttributeColumn(int columnNumber) and getSegmentColumn(int columnNumber) in Item and Section classes.

3.6.5 PriceNotFoundException

```
java.lang.Object > java.lang.Throwable > java.lang.Exception >
oracle.apps.jtf.base.resources.FrameworkException >
oracle.apps.ibe.catalog.PriceNotFoundException
```

public class **PriceNotFoundException**

extends oracle.apps.jtf.base.resources.FrameworkException

Exception class to throw when a price for an item is not found by the Pricing engine.

Used by getListPrice() and getBestPrice() in the Item and PriceObject class.

3.6.6 CatalogException

```
java.lang.Object > java.lang.Throwable > java.lang.Exception >
oracle.apps.jtf.base.resources.FrameworkException >
oracle.apps.ibe.catalog.CatalogException
```

public class **CatalogException**

extends oracle.apps.jtf.base.resources.FrameworkException

General Exception for the Catalog. Exception class to throw if error occurs while retrieving requested data.

3.6.7 SectionNotFoundException

```
java.lang.Object > java.lang.Throwable > java.lang.Exception >
oracle.apps.jtf.base.resources.FrameworkException >
oracle.apps.ibe.catalog.SectionNotFoundException
```

public class **SectionNotFoundException**

extends oracle.apps.jtf.base.resources.FrameworkException

Exception class to throw when a section is not found in the database or a section does not contain a subsection of a particular type.

Used by getLeafSubSectionIDs(), getLeafSubSections(), getRelatedSections(), getRelatedSectionIDs(), getSiblingSectionIDs(), getSiblingSections(), getSubSection(), getSubSectionID(), getSubSections(), getSubSectionIDs(), getSuperSeciton, getSuperSectionID, getSuperSections(), getSuperSectionIDs() getFeaturedSubsection(), getFeaturedSubSectionID() in the Section class.

4

Oracle iStore 11*i* Shopping Cart Quote APIs

This chapter contains the following information about the Oracle iStore 11*i* Shopping Cart Quote public class APIs in the package oracle.apps.ibe.shoppingcart.quote:

4.1 Shopping Cart Quote API Class Summary

APIs for the Oracle iStore 11*i* Shopping Cart Quote are in the package oracle.apps.ibe.shoppingcart.quote. The table below describes the classes briefly.

Table 4–1 Shopping Cart Quote Class Summary

Class	Description
Class CCTrxnOutRecord	
Class Contract	A class for Contracts.
ContractException	Exception class to throw if error occurs during Contract class method.
Class ControlRecord	Java wrapper class of the PL/SQL record type `Control_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.
Class FreightChargeRecord	Java wrapper class of the PL/SQL record type `Freight_Charge_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.
Class HeaderRecord	Java wrapper class of the PL/SQL record type `Qte_Header_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.
Class LineAttributeExtRecord	Java wrapper class of the PL/SQL record type `Line_Attribs_Ext_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.
Class LineDetailRecord	Java wrapper class of the PL/SQL record type `Qte_Line_Dtl_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.
Class LineRecord	Java wrapper class of the PL/SQL record type `Qte_Line_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.
Class LineRelationshipRecord	Java wrapper class of the PL/SQL record type `Line_Rltship_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.
Class OrderHeaderRecord	
Class PaymentRecord	Java wrapper class of the PL/SQL record type `Payment_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.
Class PriceAdjustmentAttributeRecord	Java wrapper class of the PL/SQL record type `Price_Adj_Attr_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.
Class PriceAdjustmentRecord	Java wrapper class of the PL/SQL record type `Price_Adj_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.

Table 4–1 Shopping Cart Quote Class Summary (Cont.)

Class	Description
Class PriceAdjustmentRelationshipRecord	Java wrapper class of the PL/SQL record type `Price_Adj_Rltship_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.
Class PriceAttributeRecord	Java wrapper class of the PL/SQL record type `Price_Attributes_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.
Class Quote	A class that represents quotes which are used as shopping carts, Express Checkout carts, and checked out carts.
Class QuoteAccessRecord	The sharees' access control information for quotes. The fields are based on the table `IBE_SH_QUOTE_ACCESS`.
QuoteException	Exception class to throw if Quote class method action has already been performed by others or if there is an application error.
Class ShipmentRecord	Java wrapper class of the PL/SQL record type `Shipment_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.
Class SubmitControlRecord	Java wrapper class of the PL/SQL record type `Submit_Control_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`. It is used to control the submit process.
Class TaxDetailRecord	Java wrapper class of the PL/SQL record type `Tax_Detail_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.

4.2 Class CCTrxnOutRecord

`java.lang.Object > oracle.apps.ibe.shoppingcart.quote.CCTrxnOutRecord`

public class **CCTrxnOutRecord**

extends Object

4.2.1 Variables for Class CCTrxnOutRecord

auth_code

```
public String auth_code
```

bep_err_code

```
public String bep_err_code
```

bep_err_message

```
public String bep_err_message
```

err_code

```
public String err_code
```

err_location

```
public BigDecimal err_location
```

err_message

```
public String err_message
```

nls_lang

```
public String nls_lang
```

RCS_ID

```
public static final String RCS_ID
```

status

```
public BigDecimal status
```

trxn_date

```
public Timestamp trxn_date
```

trxn_id

```
public BigDecimal trxn_id
```

4.2.2 Constructors for Class CCTrxnOutRecord

CCTrxnOutRecord

```
public CCTrxnOutRecord()
```

CCTrxnOutRecord

```
public CCTrxnOutRecord(boolean __RosettaUseGMISSValues)
```

4.3 Class Contract

`java.lang.Object > oracle.apps.ibe.shoppingcart.quote.Contract`

public class **Contract**

extends Object

A class for contracts

4.3.1 Variables for Class Contract

ACTIVE

`public static final int ACTIVE`

APPROVED

`public static final int APPROVED`

CANCELLED

`public static final int CANCELLED`

ENTERED

`public static final int ENTERED`

The possible values for the state of a contract

EXPIRED

`public static final int EXPIRED`

HOLD

`public static final int HOLD`

RCS_ID

`public static final String RCS_ID`

Standard public static final String which is initialized with the usual RCS header used by ARCS

RCS_ID_RECORDED

```
public static final boolean RCS_ID_RECORDED
```

Standard public static final boolean which is initialized by a call to oracle.apps.fnd.common.VersionInfo.recordClassVersion

SIGNED

```
public static final int SIGNED
```

TERMINATED

```
public static final int TERMINATED
```

4.3.2 Constructors for Class Contract

Contract

```
public Contract()
```

4.3.3 Methods for Class Contract

The following table is an index of Class Contract methods:

Table 4–2 Method Index for Class Contract

Method	Description
createContract	Creates a contract using the specified quote and template
getContract	Returns the contracts associated with this quote
getContractID	Retrieves the contract ID
getContractNumber	Retrieves the contract number
getContractText	Returns the text of the articles contained in this contract
getQuoteID	Retrieves the quote ID
getState	Retrieves the state of the contract possible values
notifyContractChange	Sends a notification to the specified sales representative with the comments
setEntered	Sets the state of the contract to entered
setSigned	Sets the state of the contract to signed

createContract

```
public static Contract createContract(BigDecimal quoteID, BigDecimal templateID)
throws FrameworkException, SQLException, ContractException
```

Creates a contract using the specified quote and template

getContract

```
public static Contract[] getContract(BigDecimal quoteID)
throws FrameworkException, SQLException
```

Returns the contracts associated with this quote. If no contracts are associated, then it returns null.

getContractID

```
public BigDecimal getContractID()
```

Retrieves the contract ID

Returns: the contract ID

getContractNumber

```
public String getContractNumber()
```

Retrieves the contract Number

Returns: the contract number

getContractText

```
public Reader[] getContractText()
throws FrameworkException, SQLException
```

Returns the text of the articles contained in this contract. For performance reasons, a reader stream (one for each article) is returned. After use, each reader stream should be closed by the calling application.

getContractText

```
public static Reader[] getContractText(BigDecimal contractID)
throws FrameworkException, SQLException
```

Returns the text of the articles contained in the specified contract. For performance reasons, a reader stream (one for each article) is returned. After use, each reader stream should be closed by the calling application.

getQuoteID

```
public BigDecimal getQuoteID()
```

Retrieves the quote ID

Returns: the ID of the quote associated to this contract

getState

```
public int getState()
```

Retrieves the state of the contract possible values: Contract.ENTERED, Contract.APPROVED, Contract.SIGNED, Contract.ACTIVE, Contract.CANCELLED, Contract.TERMINATED.

Returns: the state of the contract

notifyContractChange

```
public void notifyContractChange(String salesRepEmail, String comments)
throws FrameworkException, SQLException, ContractException
```

Sends a notification to the specified sales rep with the comments

setEntered

```
public void setEntered()
throws FrameworkException, SQLException, ContractException
```

Sets the state of the contract to entered

setSigned

```
public void setSigned()
throws FrameworkException, SQLException, ContractException
```

Sets the state of the contract to signed

4.4 Class ControlRecord

```
java.lang.Object > oracle.apps.ibe.shoppingcart.quote.ControlRecord
```

public class **ControlRecord**

extends Object

Java wrapper class of the PL/SQL record type `Control_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.

The fields `calculate_tax_flag` and `calculate_freight_charge_flag` can have the value "Y" for yes and "N" for no.

This class is used in methods of the class `Quote` like `appendToAndShare`, `merge`, `replaceAndShare`, and `save` for pricing.

You should not set the field `line_pricing_event` if you want to get prices for the whole quote.

4.4.1 Variables for Class ControlRecord

auto_version_flag

```
public String auto_version_flag
```

calculate_freight_charge_flag

```
public String calculate_freight_charge_flag
```

calculate_tax_flag

```
public String calculate_tax_flag
```

header_pricing_event

```
public String header_pricing_event
```

last_update_date

```
public Timestamp last_update_date
```

line_pricing_event

```
public String line_pricing_event
```

pricing_request_type

```
public String pricing_request_type
```

RCS_ID

`public static final String RCS_ID`

4.4.2 Constructors for Class ControlRecord

ControlRecord

`public ControlRecord()`

4.5 Class FreightChargeRecord

`java.lang.Object > oracle.apps.ibe.shoppingcart.quote.FreightChargeRecord`

public class **FreightChargeRecord**

extends Object

Java wrapper class of the PL/SQL record type `Freight_Charge_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.

The fields are based on the table `ASO_FREIGHT_CHARGES`.

4.5.1 Variables for Class FreightChargeRecord

attribute_category

`public String attribute_category`

attribute1

`public String attribute1`

attribute2

`public String attribute2`

attribute3

`public String attribute3`

attribute4

`public String attribute4`

attribute5

```
public String attribute5
```

attribute6

```
public String attribute6
```

attribute7

```
public String attribute7
```

attribute8

```
public String attribute8
```

attribute9

```
public String attribute9
```

attribute10

```
public String attribute10
```

attribute11

```
public String attribute11
```

attribute12

```
public String attribute12
```

attribute13

```
public String attribute13
```

attribute14

```
public String attribute14
```

attribute15

```
public String attribute15
```

charge_amount

```
public BigDecimal charge_amount
```

created_by

```
public BigDecimal created_by
```

creation_date

```
public Timestamp creation_date
```

freight_charge_id

```
public BigDecimal freight_charge_id
```

freight_charge_type_id

```
public BigDecimal freight_charge_type_id
```

last_update_date

```
public Timestamp last_update_date
```

last_update_login

```
public BigDecimal last_update_login
```

last_updated_by

```
public BigDecimal last_updated_by
```

operation_code

```
public String operation_code
```

program_application_id

```
public BigDecimal program_application_id
```

program_id

```
public BigDecimal program_id
```

program_update_date

```
public Timestamp program_update_date
```

quote_line_id

```
public BigDecimal quote_line_id
```

quote_shipment_id

`public BigDecimal quote_shipment_id`

qte_line_index

`public BigDecimal qte_line_index`

RCS_ID

`public static final String RCS_ID`

request_id

`public BigDecimal request_id`

shipment_index

`public BigDecimal shipment_index`

4.5.2 Constructors for Class FreightChargeRecord

FreightChargeRecord

`public FreightChargeRecord()`

4.6 Class HeaderRecord

`java.lang.Object > oracle.apps.ibe.shoppingcart.quote.HeaderRecord`

public class **HeaderRecord**

extends Object

Java wrapper class of the PL/SQL record type `Qte_Header_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.

The fields are based on the view `ASO_QUOTE_HEADERS_V`.

4.6.1 Variables for Class HeaderRecord

The following table is an index of Class HeaderRecord variables:

accounting_rule_id

`public BigDecimal accounting_rule_id`

attribute_category

```
public String attribute_category
```

attribute1

```
public String attribute1
```

attribute2

```
public String attribute2
```

attribute3

```
public String attribute3
```

attribute4

```
public String attribute4
```

attribute5

```
public String attribute5
```

attribute6

```
public String attribute6
```

attribute7

```
public String attribute7
```

attribute8

```
public String attribute8
```

attribute9

```
public String attribute9
```

attribute10

```
public String attribute10
```

attribute11

```
public String attribute11
```

attribute12

```
public String attribute12
```

attribute13

```
public String attribute13
```

attribute14

```
public String attribute14
```

attribute15

```
public String attribute15
```

contract_id

```
public BigDecimal contract_id
```

created_by

```
public BigDecimal created_by
```

creation_date

```
public Timestamp creation_date
```

currency_code

```
public String currency_code
```

cust_account_id

```
public BigDecimal cust_account_id
```

employee_person_id

```
public BigDecimal employee_person_id
```

exchange_rate

```
public BigDecimal exchange_rate
```

exchange_rate_date

```
public Timestamp exchange_rate_date
```

exchange_type_code

```
public String exchange_type_code
```

ffm_request_id

```
public BigDecimal ffm_request_id
```

invoice_to_address1

```
public String invoice_to_address1
```

invoice_to_address2

```
public String invoice_to_address2
```

invoice_to_address3

```
public String invoice_to_address3
```

invoice_to_address4

```
public String invoice_to_address4
```

invoice_to_city

```
public String invoice_to_city
```

invoice_to_contact_first_name

```
public String invoice_to_contact_first_name
```

invoice_to_contact_last_name

```
public String invoice_to_contact_last_name
```

invoice_to_contact_middle_name

```
public String invoice_to_contact_middle_name
```

invoice_to_country

```
public String invoice_to_country
```

invoice_to_country_code

```
public String invoice_to_country_code
```

invoice_to_county

```
public String invoice_to_county
```

invoice_to_party_id

```
public BigDecimal invoice_to_party_id
```

invoice_to_party_name

```
public String invoice_to_party_name
```

invoice_to_party_site_id

```
public BigDecimal invoice_to_party_site_id
```

invoice_to_postal_code

```
public String invoice_to_postal_code
```

invoice_to_province

```
public String invoice_to_province
```

invoice_to_state

```
public String invoice_to_state
```

invoicing_rule_id

```
public BigDecimal invoicing_rule_id
```

last_update_date

```
public Timestamp last_update_date
```

last_update_login

```
public BigDecimal last_update_login
```

last_updated_by

```
public BigDecimal last_updated_by
```

marketing_source_code

```
public String marketing_source_code
```

marketing_source_code_id

```
public BigDecimal marketing_source_code_id
```

marketing_source_name

```
public String marketing_source_name
```

order_id

```
public BigDecimal order_id
```

order_number

```
public BigDecimal order_number
```

order_type_id

```
public BigDecimal order_type_id
```

order_type_name

```
public String order_type_name
```

ordered_date

```
public Timestamp ordered_date
```

org_contact_id

```
public BigDecimal org_contact_id
```

org_id

```
public BigDecimal org_id
```

orig_mktg_source_code_id

```
public BigDecimal orig_mktg_source_code_id
```

original_system_reference

```
public String original_system_reference
```

party_id

```
public BigDecimal party_id
```

party_name

```
public String party_name
```

party_type

```
public String party_type
```

payment_amount

```
public BigDecimal payment_amount
```

person_first_name

```
public String person_first_name
```

person_last_name

```
public String person_last_name
```

person_middle_name

```
public String person_middle_name
```

phone_id

```
public BigDecimal phone_id
```

price_frozen_date

```
public Timestamp price_frozen_date
```

price_list_id

```
public BigDecimal price_list_id
```

price_list_name

```
public String price_list_name
```

program_application_id

```
public BigDecimal program_application_id
```

program_id

```
public BigDecimal program_id
```

program_update_date

```
public Timestamp program_update_date
```

qte_contract_id

```
public BigDecimal qte_contract_id
```

quote_category_code

```
public String quote_category_code
```

quote_expiration_date

```
public Timestamp quote_expiration_date
```

quote_header_id

```
public BigDecimal quote_header_id
```

quote_name

```
public String quote_name
```

quote_number

```
public BigDecimal quote_number
```

quote_password

```
public String quote_password
```

quote_source_code

```
public String quote_source_code
```

quote_status

```
public String quote_status
```

quote_status_code

```
public String quote_status_code
```

quote_status_id

```
public BigDecimal quote_status_id
```

quote_version

```
public BigDecimal quote_version
```

RCS_ID

```
public static final String RCS_ID
```

request_id

```
public BigDecimal request_id
```

sales_channel_code

```
public String sales_channel_code
```

salesrep_first_name

```
public String salesrep_first_name
```

salesrep_last_name

```
public String salesrep_last_name
```

surcharge

```
public BigDecimal surcharge
```

total_adjusted_amount

```
public BigDecimal total_adjusted_amount
```

total_adjusted_percent

```
public BigDecimal total_adjusted_percent
```

total_list_price

```
public BigDecimal total_list_price
```

total_quote_price

```
public BigDecimal total_quote_price
```

total_shipping_charge

```
public BigDecimal total_shipping_charge
```

total_tax

`public BigDecimal total_tax`

4.6.2 Constructors for Class HeaderRecord

HeaderRecord

`public HeaderRecord()`

4.7 Class LineAttributeExtRecord

`java.lang.Object > oracle.apps.ibe.shoppingcart.quote.LineAttributeExtRecord`

public class **LineAttributeExtRecord**

extends Object

Java wrapper class of the PL/SQL record type `Line_Attribs_Ext_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.

The fields are based on the table `ASO_QUOTE_LINE_ATTRIBS_EXT`.

This class stores attribute and value for quote line attributes not captured in `LineRecord` and `LineDetailRecord`.

4.7.1 Variables for Class LineAttributeExtRecord

application_id

`public BigDecimal application_id`

attribute_type_code

`public String attribute_type_code`

created_by

`public BigDecimal created_by`

creation_date

`public Timestamp creation_date`

end_date_active

```
public Timestamp end_date_active
```

last_update_date

```
public Timestamp last_update_date
```

last_update_login

```
public BigDecimal last_update_login
```

last_updated_by

```
public BigDecimal last_updated_by
```

line_attribute_id

```
public BigDecimal line_attribute_id
```

name

```
public String name
```

operation_code

```
public String operation_code
```

program_application_id

```
public BigDecimal program_application_id
```

program_id

```
public BigDecimal program_id
```

program_update_date

```
public Timestamp program_update_date
```

qte_line_index

```
public BigDecimal qte_line_index
```

quote_line_id

```
public BigDecimal quote_line_id
```

RCS_ID

`public static final String RCS_ID`

request_id

`public BigDecimal request_id`

start_date_active

`public Timestamp start_date_active`

status

`public String status`

value

`public String value`

value_type

`public String value_type`

4.7.2 Constructors for Class LineAttributeExtRecord

LineAttributeExtRecord

`public LineAttributeExtRecord()`

4.8 Class LineDetailRecord

`java.lang.Object > oracle.apps.ibe.shoppingcart.quote.LineDetailRecord`

public class **LineDetailRecord**

extends Object

Java wrapper class of the PL/SQL record type `Qte_Line_Dtl_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.

The fields are based on the table `ASO_QUOTE_LINE_DETAILS`.

Stores service-related attributes, model/option-related attributes, and return-related attributes of quote lines.

4.8.1 Variables for Class LineDetailRecord

attribute_category

`public String attribute_category`

attribute1

`public String attribute1`

attribute2

`public String attribute2`

attribute3

`public String attribute3`

attribute4

`public String attribute4`

attribute5

`public String attribute5`

attribute6

`public String attribute6`

attribute7

`public String attribute7`

attribute8

`public String attribute8`

attribute9

`public String attribute9`

attribute10

`public String attribute10`

attribute11

`public String attribute11`

attribute12

```
public String attribute12
```

attribute13

```
public String attribute13
```

attribute14

```
public String attribute14
```

attribute15

public String attribute15

change_reason_code

```
public String change_reason_code
```

complete_configuration_flag

```
public String complete_configuration_flag
```

component_code

```
public String component_code
```

config_header_id

```
public BigDecimal config_header_id
```

config_item_id

```
public BigDecimal config_item_id
```

config_revision_num

```
public BigDecimal config_revision_num
```

creation_date

```
public Timestamp creation_date
```

created_by

```
public BigDecimal created_by
```

last_update_date

```
public Timestamp last_update_date
```

last_update_login

```
public BigDecimal last_update_login
```

last_updated_by

```
public BigDecimal last_updated_by
```

operation_code

```
public String operation_code
```

program_application_id

```
public BigDecimal program_application_id
```

program_id

```
public BigDecimal program_id
```

program_update_date

```
public Timestamp program_update_date
```

qte_line_index

```
public BigDecimal qte_line_index
```

quote_line_detail_id

```
public BigDecimal quote_line_detail_id
```

quote_line_id

```
public BigDecimal quote_line_id
```

RCS_ID

```
public static final String RCS_ID
```

request_id

```
public BigDecimal request_id
```

return_attribute_category

```
public String return_attribute_category
```

return_attribute1

```
public String return_attribute1
```

return_attribute2

```
public String return_attribute2
```

return_attribute3

```
public String return_attribute3
```

return_attribute4

```
public String return_attribute4
```

return_attribute5

```
public String return_attribute5
```

return_attribute6

```
public String return_attribute6
```

return_attribute7

```
public String return_attribute7
```

return_attribute8

```
public String return_attribute8
```

return_attribute9

```
public String return_attribute9
```

return_attribute10

```
public String return_attribute10
```

return_attribute11

```
public String return_attribute11
```

return_attribute12

```
public String return_attribute12
```

return_attribute13

```
public String return_attribute13
```

return_attribute14

```
public String return_attribute14
```

return_attribute15

```
public String return_attribute15
```

return_reason_code

```
public String return_reason_code
```

return_ref_header_id

```
public BigDecimal return_ref_header_id
```

return_ref_line_id

```
public BigDecimal return_ref_line_id
```

return_ref_type

```
public String return_ref_type
```

service_coterminate_flag

```
public String service_coterminate_flag
```

service_duration

```
public BigDecimal service_duration
```

service_number

```
public BigDecimal service_number
```

service_period

```
public String service_period
```

service_ref_line_id

```
public BigDecimal service_ref_line_id
```

service_ref_line_number

```
public BigDecimal service_ref_line_number
```

service_ref_option_numb

```
public BigDecimal service_ref_option_numb
```

service_ref_order_number

```
public BigDecimal service_ref_order_number
```

service_ref_qte_line_index

```
public BigDecimal service_ref_qte_line_index
```

service_ref_shipment_numb

```
public BigDecimal service_ref_shipment_numb
```

service_ref_system_id

```
public BigDecimal service_ref_system_id
```

service_ref_type_code

```
public String service_ref_type_code
```

service_unit_list_percent

```
public BigDecimal service_unit_list_percent
```

service_unit_selling_percent

```
public BigDecimal service_unit_selling_percent
```

unit_percent_base_price

```
public BigDecimal unit_percent_base_price
```

valid_configuration_flag

```
public String valid_configuration_flag
```

4.8.2 Constructors for Class LineDetailRecord

LineDetailRecord

```
public LineDetailRecord()
```

4.9 Class LineRecord

```
java.lang.Object > oracle.apps.ibe.shoppingcart.quote.LineRecord
```

public class **LineRecord**

extends Object

Java wrapper class of the PL/SQL record type `Qte_Line_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.

The fields are based on the view `ASO_QUOTE_LINES`.

4.9.1 Variables for Class LineRecord

accounting_rule_id

```
public BigDecimal accounting_rule_id
```

attribute_category

```
public String attribute_category
```

attribute1

```
public String attribute1
```

attribute2

```
public String attribute2
```

attribute3

```
public String attribute3
```

attribute4

```
public String attribute4
```

attribute5

```
public String attribute5
```

attribute6

```
public String attribute6
```

attribute7

```
public String attribute7
```

attribute8

```
public String attribute8
```

attribute9

```
public String attribute9
```

attribute10

```
public String attribute10
```

attribute11

```
public String attribute11
```

attribute12

```
public String attribute12
```

attribute13

```
public String attribute13
```

attribute14

```
public String attribute14
```

attribute15

public String attribute15

backorder_flag

```
public String backorder_flag
```

created_by

```
public BigDecimal created_by
```

creation_date

```
public Timestamp creation_date
```

currency_code

```
public String currency_code
```

end_date_active

```
public Timestamp end_date_active
```

ffm_content_name

```
public String ffm_content_name
```

ffm_content_type

```
public String ffm_content_type
```

ffm_document_type

```
public String ffm_document_type
```

ffm_media_id

```
public String ffm_media_id
```

ffm_media_type

```
public String ffm_media_type
```

ffm_user_note

```
public String ffm_user_note
```

inventory_item_id

```
public BigDecimal inventory_item_id
```

invoice_to_party_id

```
public BigDecimal invoice_to_party_id
```

invoice_to_party_site_id

```
public BigDecimal invoice_to_party_site_id
```

invoicing_rule_id

```
public BigDecimal invoicing_rule_id
```

item_relationship_type

```
public String item_relationship_type
```

item_type_code

```
public String item_type_code
```

last_update_date

```
public Timestamp last_update_date
```

last_update_login

```
public BigDecimal last_update_login
```

last_updated_by

```
public BigDecimal last_updated_by
```

line_adjusted_amount

```
public BigDecimal line_adjusted_amount
```

line_adjusted_percent

```
public BigDecimal line_adjusted_percent
```

line_category_code

```
public String line_category_code
```

line_list_price

```
public BigDecimal line_list_price
```

line_number

```
public BigDecimal line_number
```

line_quote_price

```
public BigDecimal line_quote_price
```

marketing_source_code_id

```
public BigDecimal marketing_source_code_id
```

operation_code

```
public String operation_code
```

order_line_type_id

```
public BigDecimal order_line_type_id
```

org_id

```
public BigDecimal org_id
```

organization_id

```
public BigDecimal organization_id
```

price_list_id

```
public BigDecimal price_list_id
```

price_list_line_id

```
public BigDecimal price_list_line_id
```

pricing_quantity_uom

```
public String pricing_quantity_uom
```

program_application_id

```
public BigDecimal program_application_id
```

program_id

```
public BigDecimal program_id
```

program_update_date

```
public Timestamp program_update_date
```

quantity

`public BigDecimal quantity`

quote_header_id

`public BigDecimal quote_header_id`

quote_line_id

`public BigDecimal quote_line_id`

RCS_ID

`public static final String RCS_ID`

request_id

`public BigDecimal request_id`

related_item_id

`public BigDecimal related_item_id`

split_shipment_flag

`public String split_shipment_flag`

start_date_active

`public Timestamp start_date_active`

uom_code

`public String uom_code`

4.9.2 Constructors for Class LineRecord

LineRecord

`public LineRecord()`

4.10 Class LineRelationshipRecord

`java.lang.Object > oracle.apps.ibe.shoppingcart.quote.LineRelationshipRecord`

public class **LineRelationshipRecord**

extends Object

Java wrapper class of the PL/SQL record type `Line_Rltship_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.

The fields are based on the table `ASO_LINE_RELATIONSHIP`.

This class stores the relationship between quote lines.

4.10.1 Variables for Class LineRelationshipRecord

created_by

`public BigDecimal created_by`

creation_date

`public Timestamp creation_date`

last_update_date

`public Timestamp last_update_date`

last_update_login

`public BigDecimal last_update_login`

last_updated_by

`public BigDecimal last_updated_by`

line_relationship_id

`public BigDecimal line_relationship_id`

operation_code

`public String operation_code`

program_application_id

`public BigDecimal program_application_id`

program_id

`public BigDecimal program_id`

program_update_date

```
public Timestamp program_update_date
```

qte_line_index

```
public BigDecimal qte_line_index
```

quote_line_id

```
public BigDecimal quote_line_id
```

RCS_ID

```
public static final String RCS_ID
```

reciprocal_flag

```
public String reciprocal_flag
```

request_id

```
public BigDecimal request_id
```

related_qte_line_index

```
public BigDecimal related_qte_line_index
```

related_quote_line_id

```
public BigDecimal related_quote_line_id
```

relationship_type_code

```
public String relationship_type_code
```

4.10.2 Constructors for Class LineRelationshipRecord

LineRelationshipRecord

```
public LineRelationshipRecord()
```

4.11 Class OrderHeaderRecord

```
java.lang.Object > oracle.apps.ibe.shoppingcart.quote.OrderHeaderRecord
```

public class **OrderHeaderRecord**

extends Object

4.11.1 Variables for Class OrderHeaderRecord

contract_id

```
public BigDecimal contract_id
```

order_header_id

```
public BigDecimal order_header_id
```

order_number

```
public BigDecimal order_number
```

order_request_id

```
public BigDecimal order_request_id
```

RCS_ID

```
public static final String RCS_ID
```

status

```
public String status
```

4.11.2 Constructors for Class OrderHeaderRecord

OrderHeaderRecord

```
public OrderHeaderRecord()
```

4.12 Class PaymentRecord

```
java.lang.Object > oracle.apps.ibe.shoppingcart.quote.PaymentRecord
```

public class **PaymentRecord**

extends Object

Java wrapper class of the PL/SQL record type `Payment_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.

The fields are based on the table `ASO_PAYMENTS`.

Stores payment-related information for the quote.

4.12.1 Variables for Class PaymentRecord

attribute_category

`public String attribute_category`

attribute1

`public String attribute1`

attribute2

`public String attribute2`

attribute3

`public String attribute3`

attribute4

`public String attribute4`

attribute5

`public String attribute5`

attribute6

`public String attribute6`

attribute7

`public String attribute7`

attribute8

`public String attribute8`

attribute9

`public String attribute9`

attribute10

`public String attribute10`

attribute11

```
public String attribute11
```

attribute12

```
public String attribute12
```

attribute13

```
public String attribute13
```

attribute14

```
public String attribute14
```

attribute15

```
public String attribute15
```

created_by

```
public BigDecimal created_by
```

creation_date

```
public Timestamp creation_date
```

credit_card_approval_code

```
public String credit_card_approval_code
```

credit_card_approval_date

```
public Timestamp credit_card_approval_date
```

credit_card_code

```
public String credit_card_code
```

credit_card_expiration_date

```
public Timestamp credit_card_expiration_date
```

credit_card_holder_name

```
public String credit_card_holder_name
```

last_update_date

```
public Timestamp last_update_date
```

last_update_login

```
public BigDecimal last_update_login
```

last_updated_by

```
public BigDecimal last_updated_by
```

operation_code

```
public String operation_code
```

payment_amount

```
public BigDecimal payment_amount
```

payment_id

```
public BigDecimal payment_id
```

payment_option

```
public String payment_option
```

payment_ref_number

```
public String payment_ref_number
```

payment_term_id

```
public BigDecimal payment_term_id
```

payment_type_code

```
public String payment_type_code
```

program_application_id

```
public BigDecimal program_application_id
```

program_id

```
public BigDecimal program_id
```

program_update_date

```
public Timestamp program_update_date
```

qte_line_index

```
public BigDecimal qte_line_index
```

quote_header_id

```
public BigDecimal quote_header_id
```

quote_line_id

```
public BigDecimal quote_line_id
```

quote_shipment_id

```
public BigDecimal quote_shipment_id
```

RCS_ID

```
public static final String RCS_ID
```

request_id

```
public BigDecimal request_id
```

shipment_index

```
public BigDecimal shipment_index
```

4.12.2 Constructors for Class PaymentRecord

PaymentRecord

```
public PaymentRecord()
```

4.13 Class PriceAdjustmentAttributeRecord

```
java.lang.Object >
oracle.apps.ibe.shoppingcart.quote.PriceAdjustmentAttributeRecord
```

public class **PriceAdjustmentAttributeRecord**

extends Object

Java wrapper class of the PL/SQL record type `Price_Adj_Attr_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.

The fields are based on the view `ASO_PRICE_ADJ_ATTRIBS_V`.

4.13.1 Variables for Class PriceAdjustmentAttributeRecord

comparison_operator

```
public String comparison_operator
```

created_by

```
public BigDecimal created_by
```

creation_date

```
public Timestamp creation_date
```

flex_title

```
public String flex_title
```

last_update_date

```
public Timestamp last_update_date
```

last_update_login

```
public BigDecimal last_update_login
```

last_updated_by

```
public BigDecimal last_updated_by
```

operation_code

```
public String operation_code
```

price_adj_attrib_id

```
public BigDecimal price_adj_attrib_id
```

price_adj_index

```
public BigDecimal price_adj_index
```

price_adjustment_id

```
public BigDecimal price_adjustment_id
```

pricing_attr_value_from

```
public String pricing_attr_value_from
```

pricing_attr_value_to

```
public String pricing_attr_value_to
```

pricing_attribute

```
public String pricing_attribute
```

pricing_context

```
public String pricing_context
```

program_application_id

```
public BigDecimal program_application_id
```

program_id

```
public BigDecimal program_id
```

program_update_date

```
public Timestamp program_update_date
```

qte_line_index

```
public BigDecimal qte_line_index
```

RCS_ID

```
public static final String RCS_ID
```

request_id

```
public BigDecimal request_id
```

shipment_index

```
public BigDecimal shipment_index
```

4.13.2 Constructors for Class PriceAdjustmentAttributeRecord

PriceAdjustmentAttributeRecord

```
public PriceAdjustmentAttributeRecord()
```

4.14 Class PriceAdjustmentRecord

```
java.lang.Object > oracle.apps.ibe.shoppingcart.quote.PriceAdjustmentRecord
```

public class **PriceAdjustmentRecord**

extends Object

Java wrapper class of the PL/SQL record type `Price_Adj_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.

The fields are based on the view `ASO_PRICE_ADJUSTMENTS_V`.

4.14.1 Variables for Class PriceAdjustmentRecord

accrual_conversion_rate

```
public BigDecimal accrual_conversion_rate
```

accrual_flag

```
public String accrual_flag
```

adjusted_amount

```
public BigDecimal adjusted_amount
```

applied_flag

```
public String applied_flag
```

arithmetic_operator

```
public String arithmetic_operator
```

attribute_category

```
public String attribute_category
```

attribute1

```
public String attribute1
```

attribute2

```
public String attribute2
```

attribute3

```
public String attribute3
```

attribute4

```
public String attribute4
```

attribute5

```
public String attribute5
```

attribute6

```
public String attribute6
```

attribute7

```
public String attribute7
```

attribute8

```
public String attribute8
```

attribute9

```
public String attribute9
```

attribute10

```
public String attribute10
```

attribute11

```
public String attribute11
```

attribute12

```
public String attribute12
```

attribute13

```
public String attribute13
```

attribute14

```
public String attribute14
```

attribute15

```
public String attribute15
```

automatic_flag

```
public String automatic_flag
```

benefit_qty

```
public BigDecimal benefit_qty
```

benefit_uom_code

```
public String benefit_uom_code
```

change_reason_code

```
public String change_reason_code
```

change_reason_text

```
public String change_reason_text
```

change_sequence

```
public String change_sequence
```

charge_subtype_code

```
public String charge_subtype_code
```

charge_type_code

```
public String charge_type_code
```

cost_id

```
public BigDecimal cost_id
```

created_by

```
public BigDecimal created_by
```

creation_date

```
public Timestamp creation_date
```

credit_or_charge_flag

```
public String credit_or_charge_flag
```

estimated_flag

```
public String estimated_flag
```

expiration_date

```
public Timestamp expiration_date
```

inc_in_sales_performance

```
public String inc_in_sales_performance
```

include_on_returns_flag

```
public String include_on_returns_flag
```

invoiced_flag

```
public String invoiced_flag
```

last_update_date

```
public Timestamp last_update_date
```

last_update_login

```
public BigDecimal last_update_login
```

last_updated_by

```
public BigDecimal last_updated_by
```

list_line_no

```
public String list_line_no
```

modified_from

```
public BigDecimal modified_from
```

modified_to

```
public BigDecimal modified_to
```

modifier_header_id

```
public BigDecimal modifier_header_id
```

modifier_level_code

```
public String modifier_level_code
```

modifier_line_id

```
public BigDecimal modifier_line_id
```

modifier_line_type_code

```
public String modifier_line_type_code
```

modifier_mechanism_type_code

```
public String modifier_mechanism_type_code
```

on_invoice_flag

```
public String on_invoice_flag
```

operand

```
public BigDecimal operand
```

operation_code

```
public String operation_code
```

orig_sys_discount_ref

```
public String orig_sys_discount_ref
```

parent_adjustment_id

```
public BigDecimal parent_adjustment_id
```

price_adjustment_id

```
public BigDecimal price_adjustment_id
```

price_break_type_code

```
public String price_break_type_code
```

pricing_group_sequence

```
public BigDecimal pricing_group_sequence
```

pricing_phase_id

```
public BigDecimal pricing_phase_id
```

print_on_invoice_flag

```
public String print_on_invoice_flag
```

program_application_id

```
public BigDecimal program_application_id
```

program_id

```
public BigDecimal program_id
```

program_update_date

```
public Timestamp program_update_date
```

proration_type_code

```
public String proration_type_code
```

qte_line_index

```
public BigDecimal qte_line_index
```

quote_header_id

```
public BigDecimal quote_header_id
```

quote_line_id

```
public BigDecimal quote_line_id
```

quote_shipment_id

```
public BigDecimal quote_shipment_id
```

range_break_quantity

```
public BigDecimal range_break_quantity
```

RCS_ID

```
public static final String RCS_ID
```

rebate_payment_system_code

```
public String rebate_payment_system_code
```

rebate_transaction_reference

```
public String rebate_transaction_reference
```

rebate_transaction_type_code

```
public String rebate_transaction_type_code
```

redeemed_date

```
public Timestamp redeemed_date
```

redeemed_flag

```
public String redeemed_flag
```

request_id

```
public BigDecimal request_id
```

shipment_index

```
public BigDecimal shipment_index
```

source_system_code

```
public String source_system_code
```

split_action_code

```
public String split_action_code
```

substitution_attribute

```
public String substitution_attribute
```

tax_code

```
public String tax_code
```

tax_exempt_flag

```
public String tax_exempt_flag
```

tax_exempt_number

```
public String tax_exempt_number
```

tax_exempt_reason_code

```
public String tax_exempt_reason_code
```

update_allowable_flag

```
public String update_allowable_flag
```

update_allowed

```
public String update_allowed
```

updated_flag

```
public String updated_flag
```

4.14.2 Constructors for Class PriceAdjustmentRecord

PriceAdjustmentRecord

```
public PriceAdjustmentRecord()
```

4.15 Class PriceAdjustmentRelationshipRecord

```
java.lang.Object >
oracle.apps.ibe.shoppingcart.quote.PriceAdjustmentRelationshipRecord
```

public class **PriceAdjustmentRelationshipRecord**

extends Object

Java wrapper class of the PL/SQL record type `Price_Adj_Rltship_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.

The fields are based on the view `ASO_PRICE_ADJ_RELATIONSHIPS_V`.

Stores the relationship between quote lines and price adjustments and also between price adjustments.

4.15.1 Variables for Class PriceAdjustmentRelationshipRecord

adj_relationship_id

```
public BigDecimal adj_relationship_id
```

created_by

```
public BigDecimal created_by
```

creation_date

```
public Timestamp creation_date
```

last_update_date

```
public Timestamp last_update_date
```

last_update_login

```
public BigDecimal last_update_login
```

last_updated_by

```
public BigDecimal last_updated_by
```

operation_code

```
public String operation_code
```

price_adj_index

```
public BigDecimal price_adj_index
```

price_adjustment_id

```
public BigDecimal price_adjustment_id
```

program_application_id

```
public BigDecimal program_application_id
```

program_id

```
public BigDecimal program_id
```

program_update_date

```
public Timestamp program_update_date
```

qte_line_index

```
public BigDecimal qte_line_index
```

quote_line_id

```
public BigDecimal quote_line_id
```

quote_shipment_id

```
public BigDecimal quote_shipment_id
```

RCS_ID

```
public static final String RCS_ID
```

request_id

```
public BigDecimal request_id
```

rltd_price_adj_id

```
public BigDecimal rltd_price_adj_id
```

rltd_price_adj_index

```
public BigDecimal rltd_price_adj_index
```

shipment_index

```
public BigDecimal shipment_index
```

4.15.2 Constructors for Class PriceAdjustmentRelationshipRecord

PriceAdjustmentRelationshipRecord

```
public PriceAdjustmentRelationshipRecord()
```

4.16 Class PriceAttributeRecord

`java.lang.Object > oracle.apps.ibe.shoppingcart.quote.PriceAttributeRecord`

public class **PriceAttributeRecord**

extends Object

Java wrapper class of the PL/SQL record type `Price_Attributes_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`.

The fields are based on the table `ASO_PRICE_ATTRIBUTES`.

Stores information on qualifiers and pricing attributes for which the corresponding price adjustment line qualifies.

4.16.1 Variables for Class PriceAttributeRecord

attribute1

```
public String attribute1
```

attribute2

```
public String attribute2
```

attribute3

```
public String attribute3
```

attribute4

```
public String attribute4
```

attribute5

```
public String attribute5
```

attribute6

```
public String attribute6
```

attribute7

```
public String attribute7
```

attribute8

```
public String attribute8
```

attribute9

```
public String attribute9
```

attribute10

```
public String attribute10
```

attribute11

```
public String attribute11
```

attribute12

```
public String attribute12
```

attribute13

```
public String attribute13
```

attribute14

```
public String attribute14
```

attribute15

```
public String attribute15
```

context

```
public String context
```

created_by

```
public BigDecimal created_by
```

creation_date

```
public Timestamp creation_date
```

flex_title

```
public String flex_title
```

last_update_date

```
public Timestamp last_update_date
```

last_update_login

```
public BigDecimal last_update_login
```

last_updated_by

```
public BigDecimal last_updated_by
```

operation_code

```
public String operation_code
```

price_attribute_id

```
public BigDecimal price_attribute_id
```

pricing_attribute1

```
public String pricing_attribute1
```

pricing_attribute2

```
public String pricing_attribute2
```

pricing_attribute3

```
public String pricing_attribute3
```

pricing_attribute4

```
public String pricing_attribute4
```

pricing_attribute5

```
public String pricing_attribute5
```

pricing_attribute6

```
public String pricing_attribute6
```

pricing_attribute7

```
public String pricing_attribute7
```

pricing_attribute8

```
public String pricing_attribute8
```

pricing_attribute9

```
public String pricing_attribute9
```

pricing_attribute10

```
public String pricing_attribute10
```

pricing_attribute11

```
public String pricing_attribute11
```

pricing_attribute12

```
public String pricing_attribute12
```

pricing_attribute13

```
public String pricing_attribute13
```

pricing_attribute14

```
public String pricing_attribute14
```

pricing_attribute15

```
public String pricing_attribute15
```

pricing_attribute16

```
public String pricing_attribute16
```

pricing_attribute17

```
public String pricing_attribute17
```

pricing_attribute18

```
public String pricing_attribute18
```

pricing_attribute19

```
public String pricing_attribute19
```

pricing_attribute20

```
public String pricing_attribute20
```

pricing_attribute21

```
public String pricing_attribute21
```

pricing_attribute22

```
public String pricing_attribute22
```

pricing_attribute23

```
public String pricing_attribute23
```

pricing_attribute24

```
public String pricing_attribute24
```

pricing_attribute25

```
public String pricing_attribute25
```

pricing_attribute26

```
public String pricing_attribute26
```

pricing_attribute27

```
public String pricing_attribute27
```

pricing_attribute28

```
public String pricing_attribute28
```

pricing_attribute29

```
public String pricing_attribute29
```

pricing_attribute30

```
public String pricing_attribute30
```

pricing_attribute31

```
public String pricing_attribute31
```

pricing_attribute32

```
public String pricing_attribute32
```

pricing_attribute33

```
public String pricing_attribute33
```

pricing_attribute34

```
public String pricing_attribute34
```

pricing_attribute35

```
public String pricing_attribute35
```

pricing_attribute36

```
public String pricing_attribute36
```

pricing_attribute37

```
public String pricing_attribute37
```

pricing_attribute38

```
public String pricing_attribute38
```

pricing_attribute39

```
public String pricing_attribute39
```

pricing_attribute40

```
public String pricing_attribute40
```

pricing_attribute41

```
public String pricing_attribute41
```

pricing_attribute42

```
public String pricing_attribute42
```

pricing_attribute43

```
public String pricing_attribute43
```

pricing_attribute44

```
public String pricing_attribute44
```

pricing_attribute45

```
public String pricing_attribute45
```

pricing_attribute46

```
public String pricing_attribute46
```

pricing_attribute47

```
public String pricing_attribute47
```

pricing_attribute48

```
public String pricing_attribute48
```

pricing_attribute49

```
public String pricing_attribute49
```

pricing_attribute50

```
public String pricing_attribute50
```

pricing_attribute51

```
public String pricing_attribute51
```

pricing_attribute52

```
public String pricing_attribute52
```

pricing_attribute53

```
public String pricing_attribute53
```

pricing_attribute54

```
public String pricing_attribute54
```

pricing_attribute55

```
public String pricing_attribute55
```

pricing_attribute56
```
public String pricing_attribute56
```

pricing_attribute57
```
public String pricing_attribute57
```

pricing_attribute58
```
public String pricing_attribute58
```

pricing_attribute59
```
public String pricing_attribute59
```

pricing_attribute60
```
public String pricing_attribute60
```

pricing_attribute61
```
public String pricing_attribute61
```

pricing_attribute62
```
public String pricing_attribute62
```

pricing_attribute63
```
public String pricing_attribute63
```

pricing_attribute64
```
public String pricing_attribute64
```

pricing_attribute65
```
public String pricing_attribute65
```

pricing_attribute66
```
public String pricing_attribute66
```

pricing_attribute67
```
public String pricing_attribute67
```

pricing_attribute68

```
public String pricing_attribute68
```

pricing_attribute69

```
public String pricing_attribute69
```

pricing_attribute70

```
public String pricing_attribute70
```

pricing_attribute71

```
public String pricing_attribute71
```

pricing_attribute72

```
public String pricing_attribute72
```

pricing_attribute73

```
public String pricing_attribute73
```

pricing_attribute74

```
public String pricing_attribute74
```

pricing_attribute75

```
public String pricing_attribute75
```

pricing_attribute76

```
public String pricing_attribute76
```

pricing_attribute77

```
public String pricing_attribute77
```

pricing_attribute78

```
public String pricing_attribute78
```

pricing_attribute79

```
public String pricing_attribute79
```

pricing_attribute80

```
public String pricing_attribute80
```

pricing_attribute81

```
public String pricing_attribute81
```

pricing_attribute82

```
public String pricing_attribute82
```

pricing_attribute83

```
public String pricing_attribute83
```

pricing_attribute84

```
public String pricing_attribute84
```

pricing_attribute85

```
public String pricing_attribute85
```

pricing_attribute86

```
public String pricing_attribute86
```

pricing_attribute87

```
public String pricing_attribute87
```

pricing_attribute88

```
public String pricing_attribute88
```

pricing_attribute89

```
public String pricing_attribute89
```

pricing_attribute90

```
public String pricing_attribute90
```

pricing_attribute91

```
public String pricing_attribute91
```

pricing_attribute92

```
public String pricing_attribute92
```

pricing_attribute93

```
public String pricing_attribute93
```

pricing_attribute94

```
public String pricing_attribute94
```

pricing_attribute95

```
public String pricing_attribute95
```

pricing_attribute96

```
public String pricing_attribute96
```

pricing_attribute97

```
public String pricing_attribute97
```

pricing_attribute98

```
public String pricing_attribute98
```

pricing_attribute99

```
public String pricing_attribute99
```

pricing_attribute100

```
public String pricing_attribute100
```

pricing_context

```
public String pricing_context
```

program_application_id

```
public BigDecimal program_application_id
```

program_id

```
public BigDecimal program_id
```

program_update_date

`public Timestamp program_update_date`

qte_line_index

`public BigDecimal qte_line_index`

quote_header_id

`public BigDecimal quote_header_id`

quote_line_id

`public BigDecimal quote_line_id`

RCS_ID

`public static final String RCS_ID`

request_id

`public BigDecimal request_id`

4.16.2 Constructors for Class PriceAttributeRecord

PriceAttributeRecord

`public PriceAttributeRecord()`

4.17 Class Quote

`java.lang.Object > oracle.apps.ibe.shoppingcart.quote.Quote`

public class **Quote**

extends Object

A class that represents quotes which are used as shopping carts, Express Checkout carts, and checked out carts.

You use this class to perform operations such as creation, loading, update, deletion, and sharing of a quote, with all of its related information including quote header, quote lines, payment, shipment, tax, price adjustment, and so on. All of this information is stored in the ASO (Oracle Order Capture) tables.

All methods that perform read operations, such as `load`, `loadAll`, `loadSharees`, and `loadVersions`, execute SQL through JDBC.

All other methods that perform write operations are Java wrappers that call corresponding PL/SQL procedures through JDBC.

All of the fields of the record classes, such as HeaderRecord and LineRecord, are initialized with constant values that correspond to the PL/SQL constants `G_MISS_CHAR`, `G_MISS_NUM`, and `G_MISS_DATE>` in the PL/SQL package `FND_API`.

The following general rules apply for all methods:

- All of the optional parameters accept null value.
- The following optional parameters are included in the methods that create a quote and can be used for sharing information. If the quote is not to be shared, the parameters should be null.
  - password: the password to access the shared quote
  - url: a character string that specifies the URL to access the shared quote.

    An example:

    ```
    String url = "http://" + request.getRemoteHost() + ":"
    request.getServerPort() + "/" +
    DisplayManager.getTemplate("STORE_SCART_VIEW_LIST_P").getFileName() +
    "?minisite=" + RequestCtx.getMinisiteId() +
    "&Retrieve.x=Retrieve" +
    "&dispFlag=sharee" +
    "&sharedFlag=true" +
    "&prevref=" + DisplayManager.getTemplate("STORE_SCART_VIEW_
    LIST").getFileName() +
    "&retSharTPassword" + java.net.URLEncoder.encode(password);
    ```

  - e-mail addresses: e-mail addresses of the sharees
  - privilege types: the privilege type granted to the sharees; the valid values are:
    - 'A': all privileges
    - 'F': feedback-only privilege
    - 'R': read-only privilege
  - comment: text that is e-mailed to the sharees as a comment

- The parameter combineSameItem is used in methods such as `appendToAndShare` and `merge` that may have multiple lines for the same standard item. If the parameter is 'Y', it combines the lines into one line and updates the quantity with the sum of the quantities. The lines can be merged only if the inventory item ID and UOM code are the same, and the item type ID is standard. If the parameter is 'N', it will keep separate lines. If it is null, it will use the value of the user profile option IBE: Merge Shopping Cart Lines.
- Methods that create or update a quote have the optional parameters price list ID and currency code, which can be used to avoid the overhead of determining which price list to use to calculate the price. If you do not specify, Oracle Pricing will determine the best price list for which the user is qualified.
- Methods that create or update a quote have the optional parameter control record, to indicate pricing related flags.

4.17.1 Variables for Class Quote

COMBINED_LINES

```
public static final int COMBINED_LINES
```

Constant to have the quote to combine the lines of the same item.

controlRec

```
public ControlRecord controlRec
```

Control record used for pricing.

This class is used in methods like `appendToAndShare` , `merge`, `replaceAndShare`, and `save` for pricing.

You should not set the field line_pricing_event if you want to get prices for the whole quote.

headerFreightChargeRec

```
public FreightChargeRecord headerFreightChargeRec[]
```

Header level freight charge information.

headerPaymentRec

```
public PaymentRecord headerPaymentRec[]
```

Header level payment information.

headerPriceAttrRec

```
public PriceAttributeRecord headerPriceAttrRec[]
```

Header level price attributes.

headerRec

```
public HeaderRecord headerRec
```

Quote header information.

headerShipmentRec

```
public ShipmentRecord headerShipmentRec[]
```

Header level shipment information.

A header level shipment means the shipping information is the same for all the quote lines.

headerTaxDetailRec

```
public TaxDetailRecord headerTaxDetailRec[]
```

Header level tax information.

lineAttrExtRec

```
public LineAttributeExtRecord lineAttrExtRec[]
```

Stores attribute and value for quote line attributes not captured in lineRec and lineDetRec.

lineDetRec

```
public LineDetailRecord lineDetRec[]
```

Service related attributes, model/option related attributes, and return related attributes of quote lines.

lineFreightChargeRec

```
public FreightChargeRecord lineFreightChargeRec[]
```

Line level freight charge information.

linePaymentRec

```
public PaymentRecord linePaymentRec[]
```

Line level payment information.

linePriceAttrRec

```
public PriceAttributeRecord linePriceAttrRec[]
```

Line level price attributes.

lineRec

```
public LineRecord lineRec[]
```

Quote line information.

lineRelRec

```
public LineRelationshipRecord lineRelRec[]
```

Relationship between quote lines.

lineShipmentRec

```
public ShipmentRecord lineShipmentRec[]
```

Line level shipment information.

A line level shipment means each line has its own shipping information.

lineTaxDetailRec

```
public TaxDetailRecord lineTaxDetailRec[]
```

Line level tax information.

priceAdjAttrRec

```
public PriceAdjustmentAttributeRecord priceAdjAttrRec[]
```

Price adjustment attributes.

priceAdjRec

```
public PriceAdjustmentRecord priceAdjRec[]
```

Price adjustments.

priceAdjRelRec

```
public PriceAdjustmentRelationshipRecord priceAdjRelRec[]
```

Relationship between quote lines and price adjustments and also between price adjustments.

quoteAccessRec

```
public QuoteAccessRecord quoteAccessRec[]
```

The sharees' access control information for quotes.

RCS_ID

```
public static final String RCS_ID
```

Standard public static final String which is initialized with the usual RCS header used by ARCS.

RCS_ID_RECORDED

```
public static final boolean RCS_ID_RECORDED
```

Standard public static final boolean which is initialized by a call to oracle.apps.fnd.common.VersionInfo.recordClassVersion.

SEPARATE_LINES

```
public static final int SEPARATE_LINES
```

Constant to have the quote to have separate lines for the same item.

submitControlRec

```
public SubmitControlRecord submitControlRec
```

Control record used while submitting a quote to order.

USE_PROFILE

```
public static final int USE_PROFILE
```

Constant to let the user profile decide whether to combine the lines of the same item or to have separate lines for the same item.

4.17.2 Constructors for Class Quote

Quote

```
public Quote()
```

4.17.3 Methods for Class Quote

The following table is an index of Class Quote methods:

Table 4–3 Method Index for Class Quote

Method	Description
activate	Activates a quote, that is, makes a quote as the active quote of the user account
appendToAndShare	Appends a quote to another quote and saves sharees' information
authorizePayment	Authorizes credit card payment
delete	Deletes the quote
deleteAllLines	Delete all the lines of the quote given the quote ID
getQuoteID	Gets the quote ID given the quote number and quote version
getQuoteName	Gets the quote name given the quote number and quote version
getShareePrivilege	Gets sharee's privilege for the quote
isOrdered	Checks if the quote is ordered
isShippable	Checks if the quote is shippable
load	Loads quote information
loadAll	Loads all quotes owned by the given account excluding the active quote
loadSharees	Retrieves the quote access control information

Table 4–3 Method Index for Class Quote (Cont.)

Method	Description
loadVersions	Loads quote versions of the same quote number
merge	Merges the active quote of the guest user account to the active quote of the registered user account
replaceAndShare	Replaces a quote to another quote and saves sharee information
retrieveSharedQuote	Retrieves a shared quote
save	Creates or updates a quote
saveAsAndShare	Saves a quote and saves sharee information
share	Shares a quote with sharees
submit	Submits a quote to turn it into an order

activate

```
public static BigDecimal activate(BigDecimal quoteID, Timestamp
quoteLastUpdateDate, boolean keepOrigQuote)
throws FrameworkException, QuoteException, SQLException
```

Activates a quote, that is, makes a quote as the active quote of the user account.

Active quotes have the name IBEACTIVECART.

Parameters: quoteID - quote ID

quoteLastUpdateDate - The last update date of the quote header. Optional. If it is not null, a check is done to verify that there was no update to the quote header after the given last update date.

keepOrigQuote - Indicates if the quote of quoteID is to be kept and a new quote with the same contents as the quote of quoteID with the name IBEACTIVECART is to be created. Use true to keep the quote quoteID and create a new quote; false just to rename the quote of quoteID to IBEACTIVECART.

Returns: quote ID of the new active quote for the user account

appendToAndShare

```
public static BigDecimal appendToAndShare(BigDecimal quoteID, Timestamp
quoteLastUpdateDate, BigDecimal toQuoteID, boolean createNewVersion, int
combineSameItem, String toQuotePassword, String url, String emailAddresses[],
String privilegeTypes[], String comment, BigDecimal priceListID, String
currencyCode, ControlRecord controlRec)
throws FrameworkException, QuoteException, SQLException
```

Appends a quote to another quote and saves sharees' information.

The optional parameters toQuotePassword, url, emailAddresses, and privilegeTypes, are to share the quote.

The optional parameters priceListID and currencyCode are to specify which price list to use to calculate the price.

The optional parameter controlRec is used for pricing.

Parameters: quoteID - ID of the appending quote

quoteLastUpdateDate - Optional. The last update date of the quote header. If it is not null, a check is done to verify that there was no update to the quote header after the given last update date.

toQuoteID - ID of the quote to which the quote quoteID is appended to

createNewVersion - If true, it creates a new version of the quote toQuoteID and appends the quote quoteID to the quote toQuoteID. Otherwise, it appends the quote quoteID to the quote toQuoteID.

combineSameItem - If `COMBINED_LINES`, it combines lines of the same item. If `SEPARATE_LINES`, it creates seperate lines for the same item. If `USE_PROFILE` or any other values, it uses the user profile option IBE: Merge Shopping Cart Lines determine whether to combines lines or create seperate lines.

toQuotePassword - Optional. Password the sharees should use to access the quote toQuoteID

url - Optional. URL to access the shared quote.

emailAddresses - Optional. E-mail addresses of the sharees.

privilegeTypes - Optional. Privilege types of the sharees.

priceListID - Optional. Price list ID.

currencyCode - Optional. Currency code.

controlRec - control information for pricing

Returns: quote ID of the appended quote

authorizePayment

```
public static CCTrxnOutRecord authorizePayment(BigDecimal quoteID, Timestamp
inQuoteLastUpdateDate, BigDecimal appID, String authType, Timestamp
outQuoteLastUpdateDate)
throws FrameworkException, QuoteException, SQLException
```

Authorizes credit card payment.

The payment record of the quote should be updated with the credit card information before calling this method.

The return status is set in the QuotePmtOutRecord which contains the return values from the Payment Server. If app_id is null, the parameter defaults to the application ID of Oracle Order Capture. If authType is null, the parameter defaults to 'AUTHONLY'.

Parameters: quoteID - quote ID

inQuoteLastUpdateDate - Optional. The last update date of the quote header. If it is not null, a check is done to verify that there was no update to the quote header after the given last update date.

appID - Application ID registered in Oracle iPayment. If null, it defaults to the applcation ID of Oracle Order Capture.

authType - Authoriaztion type. 'AUTHONLY' to just authorize; 'AUTHCAPTURE' to authorize and capture

outQuoteLastUpdateDate - the last update date of the quote after authorization

Returns: record containing the result of the authorize operation

delete

```
public static void delete(BigDecimal quoteID, Timestamp quoteLastUpdateDate)
throws FrameworkException, QuoteException, SQLException
```

Deletes the quote.

Parameters: quoteID - quote ID

quoteLastUpdateDate - Optional. The last update date of the quote header. If you do not pass null, a check is done to verify that there was no update to the quote header after the given last update date.

deleteAllLines

```
public static Timestamp deleteAllLines(BigDecimal quoteID, Timestamp
quoteLastUpdateDate, BigDecimal shareeNum)
throws FrameworkException, QuoteException, SQLException
```

Delete all the lines of the quote given the quote ID

Parameters: quoteID - quote ID

quoteLastUpdateDate - Optional. The last update date of the quote header. If you do not pass null, a check is done to verify that there was no update to the quote header after the given last update date.

shareeNum - sharee number. If a user is deleting all the lines as a sharee, he/she should put his/her sharee number to see if he/she has the privilege to do so.

Returns: last update date of the quote header record

getQuoteID

```
public static BigDecimal getQuoteID(BigDecimal quoteNumber, BigDecimal
quoteVersion)
throws FrameworkException, SQLException
```

Gets the quote ID given the quote number and quote version

Parameters: quoteNumber - quote number

quoteVersion - quote version

Returns: quote ID; null if the quote ID is not found

getQuoteName

```
public static String getQuoteName(BigDecimal quoteNumber, BigDecimal
quoteVersion)
throws FrameworkException, SQLException
```

Gets the quote name given the quote number and quote version

Parameters: quoteNumber - quote number

quoteVersion - quote version

Returns: quote name; null if not found

getShareePrivilege

```
public static String getShareePrivilege(BigDecimal quoteNumber, BigDecimal
quoteVersion, BigDecimal shareeNumber, String password)
throws FrameworkException, SQLException
```

Gets sharee’s privilege for the quote. If the parameter password is not null, this method will only get the sharee’s privilege if the password is validated.

Parameters: quoteNumber - quote number

quoteVersion - quote version

shareeNumber - sharee number

password - quote password, optional

Returns: sharee's privilege

isOrdered

```
public boolean isOrdered()
```

Checks if the quote is ordered

Returns: true if the quote does not exist or it has been ordered; false otherwise

isShippable

```
public static boolean isShippable(BigDecimal quoteID)
throws FrameworkException, SQLException
```

Checks if the quote is shippable

Parameters: quoteID - quote ID

Returns: true if at least one of the line items of the qoute is shippable; false otherwise.

load

```
public static Quote load(BigDecimal quoteID, BigDecimal partyID, BigDecimal
custAcctID, boolean loadLine, boolean loadLineDetail, boolean
loadHeaderPriceAttr, boolean loadLinePriceAttr, boolean loadHeaderPayment,
boolean loadLinePayment, boolean loadHeaderShipment, boolean loadLineShipment,
boolean loadHeaderTaxDetail, boolean loadLineTaxDetail, boolean loadLineRel,
boolean loadLineAttrExt, boolean includeOrdered)
throws FrameworkException, SQLException
```

Loads quote information

headerRec field is always loaded and all the other fields are loaded depending on the boolean flags like loadLine, loadLineDetail, etc. The quote to be loaded can be an active quote, a saved quote, or a contract quote.

If quoteID is not null, it loads a quote using quoteID.

If quoteID is null, it loads the active quote using partyID and custAcctID. The following fields are not loaded.

- headerRec
  - ffm_request_id
  - qte_contract_id
  - party_name
- lineRec
  - operation_code
  - pricing_quantity_uom
  - ffm_content_name
  - ffm_document_type
  - ffm_media_type
  - ffm_media_id
  - ffm_content_type
  - ffm_user_note
- lineDetRec
  - operation_code
  - qte_line_index
  - service_ref_qte_line_index
  - return_attribute_category
  - return_reason_code
  - change_reason_code
- lineRelRec
  - operation_code
  - qte_line_index
  - related_qte_line_index
- lineAttrExtRec

  - qte_line_index
  - shipment_index
  - quote_header_id
  - quote_shipment_id
  - operation_code
- headerPaymentRec and linePaymentRec
  - operation_code
  - qte_line_index
  - shipment_index
- headerShipmentRec and lineShipmentRec
  - operation_code
  - qte_line_index
  - ship_quote_price
  - pricing_quantity
- headerTaxDetailRec and lineTaxDetailRec
  - operation_code
  - qte_line_index
  - shipment_index

Parameters: quoteID - quote ID which corresponds to the column quote_header_id in the table `ASO_QUOTE_HEADERS_ALL`

partyID - party ID

custAcctID - customer account ID

loadLine - use true to load lineRec field, false otherwise

loadLineDetail - use true to load lineDetRec field, false otherwise

loadHeaderPriceAttr - use true to load headerPriceAttrRec field, false otherwise

loadLinePriceAttr - use true to load linePriceAttrRec field, false otherwise

loadHeaderPayment - use true to load headerPaymentRec field, false otherwise

loadLinePayment - use true to load linePaymentRec field, false otherwise loadHeaderShipment - use true to load headerShipmentRec field, false otherwise

loadLineShipment - use true to load lineShipmentRec field, false otherwise

loadHeaderTaxDetail - use true to load headerTaxDetailRec field, false otherwise

loadLineTaxDetail - use true to load lineTaxDetailRec field, false otherwise

loadLineRel - use true to load lineRelRec field, false otherwise

loadLineAttrExt - use true to load lineAttrExtRec field, false otherwise

includeOrdered - use true to indicate that ordered quote can be loaded, false otherwise

Returns: quote object

loadAll

```
public static Quote[] loadAll(BigDecimal partyID, BigDecimal custAcctID, boolean
includeAllVersions, boolean includeOrdered)
throws FrameworkException, SQLException
```

Loads all quotes owned by the given account excluding the active quote. Ordered by quote number in ascending order and by quote version in descending order.

Parameters: partyID - party ID

custAcctID - customer account ID

includeAllVersions - true if all the versions of quote number should be included, false if only the latest version of quote number should be included

includeOrdered - true if the ordered quotes should be included, false otherwise

Returns: Quote objects with headerRec fields loaded.

loadSharees

```
public static QuoteAccessRecord[] loadSharees(BigDecimal quoteID)
throws FrameworkException, SQLException
```

Retrieves the quote access control information

Parameters: quoteID - quote ID

Returns: an array of QuoteAccessRecord; one record for each sharee.

loadVersions

```
public static Quote[] loadVersions(BigDecimal quoteNumber, boolean
includeOrdered)
throws FrameworkException, SQLException
```

Loads quote versions of the same quote number.

Only the field headerRec is loaded.

Parameters: quoteNumber - quote number

includeOrdered - boolean flag to indicate if the ordered quotes should be included

Returns: quote versions with the given quote number

merge

```
public static BigDecimal merge(BigDecimal guestQuoteID, Timestamp
guestQuoteLastUpdateDate, String mode, int combineSameItem, BigDecimal
regUserPartyID, BigDecimal regUserCustAcctID, BigDecimal priceListID, String
currencyCode, ControlRecord controlRec)
throws FrameworkException, QuoteException, SQLException
```

Merges the active quote of the guest user account to the active quote of the registered user account.

The optional parameters, priceListID and currencyCode are to specify which price list to use to calculate the price.

The optional parameter controlRec is used for pricing.

Parameters: guestQuoteID - ID of the appending quote

questQuoteLastUpdateDate - The last update date of the quote header. Optional parameter for concurrency control.

mode - The mode can have the value of "MERGE", "KEEP", or "REMOVE." "MERGE" is the default value and it merges the guest quote to the registered quote. "KEEP" makes the guest qutoe as the active quote in registered account. "REMOVE" removes the guest quote.

combineSameItem - If COMBINED_LINES, it combines lines of the same item. If SEPARATE_LINES, it creates seperate lines for the same item. If USE_PROFILE or any other values, it uses the user profile option IBE: Merge Shopping Cart Lines to determine whether to combine lines or create seperate lines.

priceListID - the price list ID

currencyCode - the currency code

controlRec - the control information for pricing

Returns: ID of the merged active quote of this registered account

replaceAndShare

```
public static BigDecimal replaceAndShare(BigDecimal quoteID, Timestamp
quoteLastUpdateDate, BigDecimal replacedQuoteID, boolean createNewVersion,
String replacedQuotePassword, String url, String emailAddresses[], String
privilegeTypes[], String comment, BigDecimal priceListID, String currencyCode,
ControlRecord controlRec)
throws FrameworkException, QuoteException, SQLException
```

Replaces a quote to another quote and saves sharee information.

The optional parameters, replacedQuotePassword, url, emailAddresses, and privilegeTypes, are to share the quote.

The optional parameters, priceListID and currencyCode are to specify which price list to use to calculate the price.

The optional parameter controlRec is used for pricing.

Parameters: quoteID - ID of the replacing quote

quoteLastUpdateDate - Optional. The last update date of the quote header. If you do not pass null, a check is done to verify that there was no update to the quote header after the given last update date.

replacedQuoteID - the ID of the quote by which quoteID is replaced

createNewVersion - true to create a new version of toQuoteID, false otherwise

replacedQuotePassword - the password the sharees should use to access the quote replacedQuoteID

url - URL to access the shared quote

emailAddresses - e-mail addresses of the sharees

privilegeTypes - privilege types of the sharees

priceListID - price list ID

currencyCode - currency code

controlRec - the control information for pricing

Returns: ID of the replaced quote

retrieveSharedQuote

```
public static BigDecimal retrieveSharedQuote(BigDecimal quoteNumber, BigDecimal
quoteVersion, String quotePassword, BigDecimal shareePartyID, BigDecimal
shareeCustAcctID, BigDecimal shareeNumber, BigDecimal priceListID, String
currencyCode, ControlRecord controlRec)
throws FrameworkException, QuoteException, SQLException
```

Retrieves a shared quote.

Recalculates the quote price if the user profile option IBE: Shopping Cart Price based on Owner is "N".

The optional parameters priceListID and currencyCode are to specify which price list to use to calculate the price.

The optional parameter controlRec is used for pricing.

Parameters: quoteNumber - quote number

quoteVersion - quote version

quotePassword - password to acccess the shared quote

shareePartyID - party ID of the sharee

shareeCustAcctID - customer account ID of the sharee

shareeNumber - sharee number used to find the privilege of the sharee

priceListID - Price list ID. If not null, the price engine uses this price list instead of searching for one

currencyCode - currency code

controlRec - control information for pricing

Returns: shared quote ID

save

```
public void save(BigDecimal shareePartyID, BigDecimal shareeCustAcctID,
BigDecimal shareeNumber, int combineSameItem, boolean autoUpdateActiveQuote,
boolean saveLine, boolean saveLineDetail, boolean saveHeaderPriceAttr, boolean
saveLinePriceAttr, boolean savePriceAdj, boolean savePriceAdjAttr, boolean
savePriceAdjRel, boolean saveHeaderPayment, boolean saveLinePayment, boolean
saveHeaderShipment, boolean saveLineShipment, boolean saveHeaderFreight, boolean
saveLineFreight, boolean saveHeaderTaxDetail, boolean saveLineTaxDetail, boolean
saveLineRel, boolean saveLineAttrExt)
throws FrameworkException, QuoteException, SQLException
```

Creates or updates a quote.

If headerRec.quote_header_id is not null, it is used to update the quote.

If headerRec.quote_header_id is null and headerRec.quote_name is 'IBEACTIVECART', it finds the active quote based on party ID and account ID, and updates it. If there is no current active quote, it creates a quote.

If headerRec.quote_header_id is null and headerRec.quote_name is not 'IBEACTIVECART', it creates a quote using the name headerRec.quote_name.

If the quote is a shared one, the parameters shareePartyId, shareeCustAcctId, and shareeNumber are used to check the privilege and to recalculate price based on sharee.

Parameters: combineSameItem - If COMBINED_LINES, it combines lines of the same item. If SEPARATE_LINES, it creates seperate lines for the same item. If USE_PROFILE or any other values, it uses the user profile option IBE: Merge Shopping Cart Lines to determine whether to combine lines or create separate lines. This parameter does not affect configurable and service items.

autoUpdateActiveQuote - Used only when headerRec.quote_header_id is null and there is already an active quote in the database. If true, the update to this quote is applied in the active quote. If false, QuoteException is thrown.

saveLine - use true to save lineRec, false otherwise

saveHeaderPriceAttr - use true to save headerPriceAttrRec, false otherwise

saveLinePriceAttr - use true to save linePriceAttrRec, false otherwise

savePriceAdj - use true to save priceAdjRec, false otherwise

savePriceAdjAttr - use true to save priceAdjAttrRec, false otherwise

savePriceAdjRel - use true to save priceAdjRelRec, false otherwise

saveHeaderPayment - use true to save headerPaymentRec, false otherwise

saveLinePayment - use true to save linePaymentRec, false otherwise

saveHeaderShipment - use true to save headerShipmentRec, false otherwise

saveLineShipment - use true to save lineShipmentRec, false otherwise

saveHeaderFreight - use true to save headerFreightChargeRec, false otherwise

saveLineFreight - use true to save lineFreightChargeRec, false otherwise

saveHeaderTaxDetail - use true to save headerTaxDetailRec, false otherwise

saveLineTaxDetail - use true to save lineTaxDetailRec, false otherwise saveLineDetail - use true to save lineDetRec, false otherwise

saveLineRel - use true to save lineRelRec, false otherwise

saveLineAttrExt - use true to save lineAttrExtRec, false otherwise

saveAsAndShare

```
public static BigDecimal saveAsAndShare(BigDecimal quoteID, Timestamp
quoteLastUpdateDate, String newQuoteName, String newQuoteSourceCode, BigDecimal
partyID, BigDecimal custAcctID, String newQuotePassword, String url, String
emailAddresses[], String privilegeTypes[], String comment, BigDecimal
priceListID, String currencyCode, ControlRecord controlRec)
throws FrameworkException, QuoteException, SQLException
```

Saves a quote and saves sharee information.

The optional parameters newQuotePassword, url, emailAddresses, and privilegeTypes, are to share the quote.

The optional parameters priceListID and currencyCode are to specify which price list to use to calculate the price.

The optional parameter controlRec is used for pricing.

Parameters: quoteID - the quote ID

quoteLastUpdateDate - Optional. The last update date of the quote header. If you do not pass null, a check is done to verify that there was no update to the quote header after the given last update date.

newQuoteName - the new quote name

newQuoteSourceCode - the new quote source code

partyID - party ID

custAcctID - customer account ID

newQuotePassword - password to access new quote as a sharee

url - URL to access the shared quote

emailAddresses - e-mail addresses of the sharees

privilegeTypes - privilege types of the sharees

priceListID - price list ID

currencyCode - currency code

controlRec - control information for pricing

Returns: quote ID of the saved quote

share

```
public void share(BigDecimal quoteID, String url, String emailAddresses[],
String privilegeTypes[], String comment)
throws FrameworkException, QuoteException, SQLException
```

Shares a quote with sharees

Parameters: quoteID - quote ID

url - URL

emailAddresses - e-mail addresses, one for each sharee

privilegeTypes - privilege types, one for each sharee

submit

```
public static OrderHeaderRecord submit(BigDecimal quoteID, Timestamp
quoteLastUpdateDate, String salesRepAssistCode, String salesRepEmailAddr, String
commentForSalesRep, BigDecimal shareePartyID, BigDecimal shareeCustAcctID,
BigDecimal shareeNumber, SubmitControlRecord submitControlRec)
throws FrameworkException, QuoteException, SQLException
```

Submits a quote to turn it into an order.

The parameters salesRepAssistCode, salesRepEmailAddr, and commentForSalesRep are used when the customer wants to send an e-mail to the sales representative with a comment.

The parameters shareePartyID, shareeCustAccntID, and shareeNumber need to be passed when a sharee wants to submit a quote.

Parameters: quoteID - quote ID

quoteLastUpdateDate - Optional. The last update date of the quote header. If you do not pass null, a check is done to verify that there was no update to the quote header after the given last update date.

salesRepAssistCode - Optional. The string that contains the reason code for the assistance of sales representative.

salesRepEmailAddr - Optional. The e-mail address of the sales representative.

commentForSalesRep - Optional. The comment for the sales representative.

shareePartyID - Optional. The party ID of the sharee who submits the quote.

shareeCustAcctID - Optional. The customer account ID of the sharee who submits the quote.

shareeNumber - Optional. The sharee number of the sharee who submits the quote.

submitControlRec - Optional. Submit control information that includes book flag, reserve flag, calculate price flag, and server ID. If null, the default values of 'F', 'F', 'F', and '-1' are used.

Returns: order header information that includes order number, order header ID, order request ID, contract ID, and status

4.18 Class QuoteAccessRecord

```
java.lang.Object > oracle.apps.ibe.shoppingcart.quote.QuoteAccessRecord
```

public class **QuoteAccessRecord**

extends Object

The sharees' access control information for quotes. The fields are based on the table `IBE_SH_QUOTE_ACCESS`.

4.18.1 Variables for Class QuoteAccessRecord

created_by

```
public BigDecimal created_by
```

creation_date

```
public Timestamp creation_date
```

email_contact_address

```
public String email_contact_address
```

last_update_date

```
public Timestamp last_update_date
```

last_update_login

```
public BigDecimal last_update_login
```

last_updated_by

`public BigDecimal last_updated_by`

object_version_number

`public BigDecimal object_version_number`

program_application_id

`public BigDecimal program_application_id`

program_id

`public BigDecimal program_id`

program_update_date

`public Timestamp program_update_date`

quote_header_id

`public BigDecimal quote_header_id`

quote_sharee_id

`public BigDecimal quote_sharee_id`

quote_sharee_number

`public BigDecimal quote_sharee_number`

RCS_ID

`public static final String RCS_ID`

request_id

`public BigDecimal request_id`

update_privilege_type_code

`public String update_privilege_type_code`

4.18.2 Constructors for Class QuoteAccessRecord

QuoteAccessRecord

`public QuoteAccessRecord()`

4.19 Class ShipmentRecord

`java.lang.Object > oracle.apps.ibe.shoppingcart.quote.ShipmentRecord`

public class **ShipmentRecord**

extends Object

Java wrapper class of the PL/SQL record type Shipment_Rec_Type in the PL/SQL package ASO_QUOTE_PUB.

The fields are based on the view ASO_SHIPMENTS_V.

Stores shipping information for a quote at header or line level.

4.19.1 Variables for Class ShipmentRecord

attribute_category

```
public String attribute_category
```

attribute1

```
public String attribute1
```

attribute2

```
public String attribute2
```

attribute3

```
public String attribute3
```

attribute4

```
public String attribute4
```

attribute5

```
public String attribute5
```

attribute6

```
public String attribute6
```

attribute7

```
public String attribute7
```

attribute8

```
public String attribute8
```

attribute9

```
public String attribute9
```

attribute10

```
public String attribute10
```

attribute11

```
public String attribute11
```

attribute12

```
public String attribute12
```

attribute13

```
public String attribute13
```

attribute14

```
public String attribute14
```

attribute15

```
public String attribute15
```

created_by

```
public BigDecimal created_by
```

creation_date

```
public Timestamp creation_date
```

fob_code

```
public String fob_code
```

freight_terms_code

```
public String freight_terms_code
```

freight_carrier_code

```
public String freight_carrier_code
```

last_update_date

```
public Timestamp last_update_date
```

last_update_login

```
public BigDecimal last_update_login
```

last_updated_by

```
public BigDecimal last_updated_by
```

operation_code

```
public String operation_code
```

order_line_id

```
public BigDecimal order_line_id
```

packing_instructions

```
public String packing_instructions
```

pricing_quantity

```
public BigDecimal pricing_quantity
```

program_application_id

```
public BigDecimal program_application_id
```

program_id

```
public BigDecimal program_id
```

program_update_date

```
public Timestamp program_update_date
```

promise_date

```
public Timestamp promise_date
```

qte_line_index

```
public BigDecimal qte_line_index
```

quantity

```
public BigDecimal quantity
```

quote_header_id

```
public BigDecimal quote_header_id
```

quote_line_id

```
public BigDecimal quote_line_id
```

RCS_ID

```
public static final String RCS_ID
```

request_date

```
public Timestamp request_date
```

request_id

```
public BigDecimal request_id
```

reservation_id

```
public BigDecimal reservation_id
```

reserved_quantity

```
public BigDecimal reserved_quantity
```

schedule_ship_date

```
public Timestamp schedule_ship_date
```

ship_method_code

```
public String ship_method_code
```

ship_quote_price

```
public BigDecimal ship_quote_price
```

ship_partial_flag

```
public String ship_partial_flag
```

ship_set_id

```
public BigDecimal ship_set_id
```

ship_to_address1

```
public String ship_to_address1
```

ship_to_address2

```
public String ship_to_address2
```

ship_to_address3

```
public String ship_to_address3
```

ship_to_address4

```
public String ship_to_address4
```

ship_to_city

```
public String ship_to_city
```

ship_to_contact_first_name

```
public String ship_to_contact_first_name
```

ship_to_contact_last_name

```
public String ship_to_contact_last_name
```

ship_to_contact_middle_name

```
public String ship_to_contact_middle_name
```

ship_to_country

```
public String ship_to_country
```

ship_to_country_code

```
public String ship_to_country_code
```

ship_to_county

```
public String ship_to_county
```

ship_to_party_id

```
public BigDecimal ship_to_party_id
```

ship_to_party_name

```
public String ship_to_party_name
```

ship_to_party_site_id

```
public BigDecimal ship_to_party_site_id
```

ship_to_postal_code

```
public String ship_to_postal_code
```

ship_to_province

```
public String ship_to_province
```

ship_to_state

```
public String ship_to_state
```

shipment_id

```
public BigDecimal shipment_id
```

shipment_priority_code

```
public String shipment_priority_code
```

shipping_instructions

```
public String shipping_instructions
```

4.19.2 Constructors for Class ShipmentRecord

ShipmentRecord

```
public ShipmentRecord()
```

4.20 Class SubmitControlRecord

`java.lang.Object > oracle.apps.ibe.shoppingcart.quote.SubmitControlRecord`

public class **SubmitControlRecord**

extends Object

Java wrapper class of the PL/SQL record type `Submit_Control_Rec_Type` in the PL/SQL package `ASO_QUOTE_PUB`. It is used to control the submit process.

It has the following 4 fields:

- book_flag: "F" is the default value
- reserve_flag: "F" is the default value
- calculate_price: "F" is the default value
- server_id: -1 is the default value

The `book_flag`, `reserve_flag`, and `calculate_price` fields can have the values "T" for true and "F" for false.

4.20.1 Variables for Class SubmitControlRecord

book_flag

```
public String book_flag
```

calculate_price

```
public String calculate_price
```

RCS_ID

```
public static final String RCS_ID
```

reserve_flag

```
public String reserve_flag
```

server_id

```
public BigDecimal server_id
```

4.20.2 Constructors for Class SubmitControlRecord

SubmitControlRecord

`public SubmitControlRecord()`

4.21 Class TaxDetailRecord

`java.lang.Object > oracle.apps.ibe.shoppingcart.quote.TaxDetailRecord`

public class **TaxDetailRecord**

extends Object

Java wrapper class of the PL/SQL record type Tax_Detail_Rec_Type in the PL/SQL package ASO_QUOTE_PUB.

The fields are based on the table `ASO_TAX_DETAILS`.

4.21.1 Variables for Class TaxDetailRecord

attribute_category

`public String attribute_category`

attribute1

`public String attribute1`

attribute2

`public String attribute2`

attribute3

`public String attribute3`

attribute4

`public String attribute4`

attribute5

`public String attribute5`

attribute6

```
public String attribute6
```

attribute7

```
public String attribute7
```

attribute8

```
public String attribute8
```

attribute9

```
public String attribute9
```

attribute10

```
public String attribute10
```

attribute11

```
public String attribute11
```

attribute12

```
public String attribute12
```

attribute13

```
public String attribute13
```

attribute14

```
public String attribute14
```

attribute15

```
public String attribute15
```

created_by

```
public BigDecimal created_by
```

creation_date

```
public Timestamp creation_date
```

last_update_date

```
public Timestamp last_update_date
```

last_update_login

```
public BigDecimal last_update_login
```

last_updated_by

```
public BigDecimal last_updated_by
```

operation_code

```
public String operation_code
```

orig_tax_code

```
public String orig_tax_code
```

program_application_id

```
public BigDecimal program_application_id
```

program_id

```
public BigDecimal program_id
```

program_update_date

```
public Timestamp program_update_date
```

qte_line_index

```
public BigDecimal qte_line_index
```

quote_header_id

```
public BigDecimal quote_header_id
```

quote_line_id

```
public BigDecimal quote_line_id
```

quote_shipment_id

```
public BigDecimal quote_shipment_id
```

RCS_ID

```
public static final String RCS_ID
```

request_id

```
public BigDecimal request_id
```

shipment_index

```
public BigDecimal shipment_index
```

tax_amount

```
public BigDecimal tax_amount
```

tax_code

```
public String tax_code
```

tax_date

```
public Timestamp tax_date
```

tax_detail_id

```
public BigDecimal tax_detail_id
```

tax_exempt_flag

```
public String tax_exempt_flag
```

tax_exempt_number

```
public String tax_exempt_number
```

tax_exempt_reason_code

```
public String tax_exempt_reason_code
```

tax_rate

```
public BigDecimal tax_rate
```

4.21.2 Constructors for Class TaxDetailRecord

TaxDetailRecord

```
public TaxDetailRecord()
```

4.22 Exception Classes

The methods in the package oracle.apps.ibe.shoppingcart.quote throw the following exceptions:

- SQLException
- FrameworkException
- ContractException
- QuoteException

4.22.1 SQLException

Exception class to throw if database error occurs.

4.22.2 FrameworkException

Exception class to throw if error occurs while trying to get connection.

4.22.3 ContractException

```
java.lang.Object > java.lang.Throwable > java.lang.Exception >
oracle.apps.jtf.base.resources.FrameworkException >
oracle.apps.ibe.shoppingcart.quote.ContractException
```

public class **ContractException**

extends FrameworkException

Exception class to throw if error occurs during Contract class method.

Variables for Class ContractException

RCS_ID

```
public static final String RCS_ID
```

RCS_ID_RECORDED

```
public static final boolean RCS_ID_RECORDED
```

Constructors for Class ContractException

ContractException

```
public ContractException(String errorKey)
```

Construct an exception with the errorKey.

ContractException

```
public ContractException(String errorKey, Object params[])
```

Construct an exception with the errorKey.

Parameters: params - an array of tokens for errorKey

ContractException

```
public ContractException(Exception e, String errorKey, Object params[])
```

Construct an Exception with the given exception and errorKey.

Parameters: e - the parent exception

params - an array of tokens for errorKey

ContractException

```
public ContractException(Exception e, String errorKey, String param)
```

ContractException

```
public ContractException(Exception e, String errorKey)
```

ContractException

```
public ContractException(String err_msg, String errorKey, Object params[])
```

Construct an Exception with the errorKey and error message.

Parameters: params - an array of tokens for errorKey

ContractException

```
public ContractException(String err_msg, String errorKey, String param)
```

ContractException

```
public ContractException(String err_msg, String errorKey)
```

ContractException

```
public ContractException(int err_msg_count, String errorKey, Object params[])
throws FrameworkException
```

Construct an Exception with the errorKey and errors occurred at PL/SQL level.

Parameters: err_msg_count - the number of messages to be returned from the PL/SQL error stack

params - an array of tokens for the errorKey

ContractException

```
public ContractException(int err_msg_count, String errorKey, String param)
throws FrameworkException
```

ContractException

```
public ContractException(int err_msg_count, String errorKey)
throws FrameworkException
```

4.22.4 QuoteException

```
java.lang.Object > java.lang.Throwable > java.lang.Exception >
oracle.apps.jtf.base.resources.FrameworkException >
oracle.apps.ibe.shoppingcart.quote.QuoteException
```

public class **QuoteException**

extends FrameworkException

Exception class to throw if Quote class method action has already been performed by others or if there is an application error.

Variables for Class QuoteException

RCS_ID

```
public static final String RCS_ID
```

RCS_ID_RECORDED

```
public static final boolean RCS_ID_RECORDED
```

Constructors for Class QuoteException

QuoteException

```
public QuoteException(String errorKey)
```

Construct an exception with the errorKey.

QuoteException

```
public QuoteException(String errorKey, Object params[])
```

Construct an exception with the errorKey.

Parameters: params - an array of tokens for errorKey

QuoteException

```
public QuoteException(Exception e, String errorKey, Object params[])
```

Construct an Exception with the given exception and errorKey.

Parameters: e - the parent exception

params - an array of tokens for errorKey

QuoteException

```
public QuoteException(Exception e, String errorKey, String param)
```

QuoteException

```
public QuoteException(Exception e, String errorKey)
```

QuoteException

```
public QuoteException(String err_msg, String errorKey, Object params[])
```

Construct an Exception with the errorKey and error message.

Parameters: params - an array of tokens for errorKey

QuoteException

```
public QuoteException(String err_msg, String errorKey, String param)
```

QuoteException

```
public QuoteException(String err_msg, String errorKey)
```

QuoteException

```
public QuoteException(int err_msg_count, String errorKey, Object params[])
throws FrameworkException
```

Construct an Exception with the errorKey and errors occurred at PL/SQL level.

Parameters: err_msg_count - the number of messages to be returned from the PL/SQL error stack

params - an array of tokens for the errorKey

QuoteException

```
public QuoteException(int err_msg_count, String errorKey, String param)
throws FrameworkException
```

QuoteException

```
public QuoteException(int err_msg_count, String errorKey)
throws FrameworkException
```

5

Oracle iStore 11*i* Postsales APIs

This chapter contains the following information on the Oracle iStore 11*i* Postsales public class APIs in the package oracle.apps.ibe.postsales:

5.1 Postsales API Class Summary

APIs for the Oracle iStore 11*i* Postsales procedures are located in the package oracle.apps.ibe.postsales. The table below describes the classes briefly.

Table 5–1 Postsales Class Summary

Class Name	Description
Class AkCurrencyFormatter	AkCurrencyFormatter implements the QueryFormatter interface for AkQuery objects.
Class AkDateFormatter	AkDateFormatter implements the QueryFormatter interface for AkQuery objects.
Class AkQuery	The AkQuery class extends the Query abstract class based on AK regions.
Class AkQueryCondition	AkQueryCondition encapsulates a single condition in the WHERE clause of an @see AkQuery.
Class AkRegion	The AkRegion class encapsulates AK regions for Postsales (e.g. Order Tracker) queries.
Class IbeAtpPvt	IbeAtpPvt is a Rosetta-generated Java wrapper for the `IBE_ATP_PVT` PL/SQL package.
Class Query	The Query abstract class specifies the minimum functionality required for database queries on which the iStore Postsales pages (e.g., Order Tracker) depend.
Class QueryCondition	QueryCondition encapsulates a single condition in the WHERE clause of a @see Query.
Class QueryUtil	The QueryUtil class contains basic utility methods for Postsales, such as null handling methods, date validation methods, and methods to check if a given value is a valid number.
Class QueryValidatorException	Exception class that extends the FrameworkException class and is thrown by all Postsales interaction query methods.

5.2 Class AkCurrencyFormatter

```
java.lang.Object > oracle.apps.ibe.postsales.AkCurrencyFormatter
```

public final class **AkCurrencyFormatter**

extends Object

implements QueryFormatter

AkCurrencyFormatter implements the QueryFormatter interface for AkQuery objects.

See Also: QueryFormatter, AkQuery

5.2.1 Variables for Class AkCurrencyFormatter

RCS_ID

```
public static final String RCS_ID
```

RCS_ID_RECORDED

```
public static final boolean RCS_ID_RECORDED
```

thisClass

```
public static final String thisClass
```

5.2.2 Methods for Class AkCurrencyFormatter

The following table is an index of Class AkCurrencyFormatter methods:

Table 5–2 Method Index for Class AkCurrencyFormatter

Method	Description
format	Implements the format method of the QueryFormatter interface
getAkCurrencyFormatter	Factory method for AkCurrencyFormatter objects
getItemName	Returns the name of the region item which holds the amount to be formatted

format

```
public final String format(ResultSet rs)
throws SQLException, FrameworkException
```

Implements the format method of the QueryFormatter interface. Given a JDBC result set, formats the amount in the amount column according to the currency code in the currency column (if specified), or the global currency code. Returns a string containing the formatted amount. Uses the formatNumber method of the oracle.apps.ibe.catalog.PriceObject class to do the work.

Parameters: rs - result set containing the row with the amount to be formatted

Returns: a string containing the formatted amount (or null if amount is null)

Throws: SQLException if a database exception occurs while fetching from the result set

Throws: FrameworkException if the amount cannot be formatted

getAkCurrencyFormatter

```
public static final AkCurrencyFormatter getAkCurrencyFormatter(Query q, String
currencyItemName, String amountItemName)
throws FrameworkException
```

Factory method for AkCurrencyFormatter objects. Returns a formatter object that formats the amount in the specified region item using the currency code in the specified region item. If such a formatter already exists and the object cache is enabled, returns the cached object; otherwise creates a new object.

Parameters: q - query whose results are to be formatted

currencyItemName - name of region item specifying the currency code

amountItemName - name of region item containing the amount

Returns: an AkCurrencyFormatter object (possibly a cached copy)

Throws: FrameworkException if either region item name is invalid

getAkCurrencyFormatter

```
public static final AkCurrencyFormatter getAkCurrencyFormatter(Query q, String
amountItemName)
throws FrameworkException
```

Factory method for AkCurrencyFormatter objects. Returns a formatter object that formats the amount in the specified region item using the currency code from the cookie. If such a formatter already exists and the object cache is enabled, returns the cached object; otherwise creates a new object. Note that this method will only provide useful formatter objects in a single- currency store where the currency code stored in the cookie is valid for all amounts displayed.

Parameters: q - query whose results are to be formatted

amountItemName - name of region item containing the amount

Returns: an AkCurrencyFormatter object (possibly a cached copy)

Throws: FrameworkException if the amount item name is invalid

getItemName

`public final String getItemName()`

Returns the name of the region item which holds the amount to be formatted.

Returns: name of AK region item containing the amount to be formatted

5.3 Class AkDateFormatter

`java.lang.Object > oracle.apps.ibe.postsales.AkDateFormatter`

public class **AkDateFormatter**

extends Object

implements QueryFormatter

AkDateFormatter implements the QueryFormatter interface for AkQuery objects.

See Also: QueryFormatter, AkQuery

5.3.1 Variables for Class AkDateFormatter

RCS_ID

`public static final String RCS_ID`

RCS_ID_RECORDED

`public static final boolean RCS_ID_RECORDED`

thisClass

`public static final String thisClass`

5.3.2 Methods for Class AkDateFormatter

The following table is an index of Class AkDateFormatter methods:

Table 5–3 Method Index for Class AkDateFormatter

Method	Description
format	Implements the format method of the QueryFormatter interface
getAkDateFormatter	Factory method for AkDateFormatter objects

Table 5–3 Method Index for Class AkDateFormatter (Cont.)

Method	Description
getItemName	Returns the name of the region item which holds the date to be formatted

format

```
public final String format(ResultSet rs)
throws SQLException, FrameworkException
```

Implements the format method of the QueryFormatter interface. Given a JDBC result set, formats the date in the date column according to the stored date format mask. Returns a string containing the formatted date.

Parameters: rs - result set containing the row with the date to be formatted

Returns: a string containing the formatted date (or null if date is null)

Throws: SQLException if a database exception occurs while fetching from the result set

Throws: FrameworkException if the date cannot be formatted

getAkDateFormatter

```
public static final AkDateFormatter getAkDateFormatter(Query q, String
dateItemName)
throws FrameworkException
```

Factory method for AkDateFormatter objects. Returns a formatter object that formats the date in the specified region item using the default date format from the cookie. If such a formatter already exists and the object cache is enabled, returns the cached object; otherwise creates a new object.

Parameters: q - query whose results are to be formatted

dateItemName - name of region item containing the date

Returns: an AkDateFormatter object (possibly a cached copy)

Throws: FrameworkException if either region item name is invalid

getItemName

```
public final String getItemName()
```

Returns the name of the region item which holds the date to be formatted.

Returns: name of AK region item containing the date to be formatted

5.4 Class AkQuery

```
java.lang.Object > oracle.apps.ibe.postsales.Query >
oracle.apps.ibe.postsales.AkQuery
```

public final class **AkQuery**

extends Query

The AkQuery class extends the Query abstract class based on AK regions.

Oracle iStore 11*i*'s Order Tracker uses the AK regions for storing the meta-data related to the display of the Orders, Invoices, Shipments and Payments information. There are corresponding Java objects that retrieve the information from these AK regions and display them in Oracle iStore 11*i* JSPs. All of the Java objects are in the `$IBE_TOP/java/postsales` directory.

AkQuery.java is the main Java object used by all JSPs.

See Also: QueryFormatter, QueryValidator, QueryUtil

5.4.1 Variables for Class AkQuery

RCS_ID

```
public static final String RCS_ID
```

RCS_ID_RECORDED

```
public static final boolean RCS_ID_RECORDED
```

thisClass

```
public static final String thisClass
```

5.4.2 Constructors for Class AkQuery

AkQuery

```
public AkQuery()
```

Constructor.

5.4.3 Methods for Class AkQuery

The following table is an index of Class AkQuery methods:

Table 5–4 Method Index for Class AkQuery

Method	Description
addCondition	Adds a condition to the WHERE clause of the query
addFormatter	Registers a QueryFormatter object for the specified query item
connect	Connects to the database and initializes the named query
disconnect	Disconnects from the database and cleans up the query
execute	Executes the query
fromURL	Reconstructs the state of a query from URL parameters generated by an HTML form or a call to toURL()
getBatchSize	Returns the maximum number of records to be displayed on a single page
getColumnIndex	Returns the index of the database view column on which the named query item is based
getConditions	Returns a vector of all QueryCondition objects added to the query via the addCondition() methods
getDateFormat	Returns the Oracle date format mask (e.g. DD-MON-YYYY) used by the query
getInvoiceId	
getItemIndex	Returns the index of the query item (i.e. AK region item or view column) with the specified name
getItemLabel	Returns the display label of the query item with the specified parameter, suitable for use as a column heading when rendering the query result table on the page
getItemName	Returns the name of the query item (i.e. AK region item or view column) with the specified index
getName	Returns a string that uniquely identifies the region (along with the responsibility ID, application ID, and language code used when connecting to the region)
getNumItems	Returns the number of items (i.e. fields) in a record returned by the query
getNumRowsFetched	Returns the total number of rows fetched by the last call to execute()

Table 5–4 Method Index for Class AkQuery (Cont.)

Method	Description
getNumRowsShown	Returns the actual number of records to be displayed on the current page
getStartRow	Returns the index of the first row to be displayed on the current page
getValue	Returns the string representation of the value in the specified cell of the query result table
isDisplayable	Returns true if the specified query item may be displayed
isQueryable	Returns true if the specified query item may be searched on
isSelectable	Returns true if the specified query item is based on a database object and thus may be included in a SELECT or ORDER BY clause
resetConditions	Removes all conditions from the WHERE clause of the query
setBatchSize	Sets the batch size (i.e. the maximum number of records shown per page) to the specified value
setOrderByColumn	Sets the ORDER BY clause of the query to use the specified item and sort direction
setStartRow	Sets the start row to the specified row for execution
showQueryConditions	Returns a properly formatted HTML option list of search conditions
showQueryOperators	Returns a properly formatted HTML option list of search operators
toURL	Externalizes the current state of the query as a URL parameter string of the form "param1=val1¶m2=val2&...¶mX=valX"

addCondition

```
public final void addCondition(int itemIndex, String operatorCode, String value)
throws FrameworkException
```

Adds a condition to the WHERE clause of the query. Successive calls to addCondition() add new conditions to the WHERE clause in order. The condition is of the form CUST_ACCOUNT_ID = 1001. The specified item must be one for which isSelectable() returns true. The specified operator must be one of the valid operators returned by showQueryOperators(). For queries based on AK regions, valid operators are AIS, BNOT, CCONTAIN, DSTART, EEND, FGREATER, and GLESS.

The value may be any free-form String, but it must be convertible to the specified item's datatype. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - index of query item

operatorCode - valid binary operator code

value - value against which item is to be compared

Throws: FrameworkException if itemIndex is invalid

Overrides: addCondition in class Query

See Also: addCondition, isSelectable, showQueryOperators, connect

addCondition

```
public final void addCondition(String itemName, String operatorCode, String
value)
throws FrameworkException
```

Adds a condition to the WHERE clause of the query. Successive calls to addCondition() add new conditions to the WHERE clause in order. The condition is of the form `<itemName> <operator> <value>`, e.g. CUST_ACCOUNT_ID = 1001. The specified item must be one for which isSelectable() returns true. The specified operator must be one of the valid operators returned by showQueryOperators(). For queries based on AK regions, valid operators are AIS, BNOT, CCONTAIN, DSTART, EEND, FGREATER, and GLESS. The value may be any free-form String, but it must be convertible to the specified item's datatype. If this method is called before connect(), the result is undefined.

Parameters: itemName - name of query item

operatorCode - valid binary operator code

value - value against which item is to be compared

Throws: FrameworkException if itemName is not found

Overrides: addCondition in class Query

See Also: addCondition, isSelectable, showQueryOperators, connect

Example To add a where clause with Cust_account_id = Customer ID from the cookie, for the Order Summary page, you can invoke the addCondition method as follows (assume q as an instantiated object of AkQuery):

```
q.addCondition("IBE_OS_CUST_ACCOUNT_ID", "AIS",
RequestCtx.getAccountId().toString());
```

where IBE_OS_CUST_ACCOUNT_ID is the region item name which represents Cust_account_id.

addCondition

```
public final void addCondition(String condition)
```

Appends the specified string to the WHERE clause of the query as is, without performing any syntactic or semantic verification. The programmer must ensure that the argument is a valid condition given the specifics of the query. If this method is called before connect(), the result is undefined.

Parameters: condition - any valid condition to be appended to the WHERE clause

Overrides: addCondition in class Query

See Also: addCondition, addCondition, connect

Example An additional column has been added to the view and there is no AK region item specified for that column. If you want to use the column in the where clause, you can use the above method. There is a column called "Order_category_code" in the view IBE_ORDER_SUM_V for which an AK region item has not been defined. If you want to use the Order_category_code in the where clause, then you can invoke the addCondition method as follows (assume q as an instantiated object of AkQuery):

```
q.addCondition("Order_category_code = 'RETURN' ");
```

addFormatter

```
public final void addFormatter(QueryFormatter formatter)
throws FrameworkException
```

Registers a QueryFormatter object for the specified query item. A QueryFormatter object could be a DateFormatter or CurrencyFormatter. For more details, refer to the the API documentation for the QueryFormatter, AkDateFormatter and AkCurrencyFormatter classes. As it fetches data from the database, the execute() method invokes the appropriate QueryFormatter objects to format specific query items in the desired way.

Parameters: formatter - the QueryFormatter object to be registered

itemIndex - index of the query item to be formatted

Overrides: addFormatter in class Query

Example Currency Formatting: IBE_OH_ORDER_TOTAL is an AK region item that corresponds to the amount column, which requires currency formatting. You can invoke the addFormatter method as follows (assume q as an instantiated object of AkQuery):

```
q.addFormatter(AkCurrencyFormatter.getAkCurrencyFormatter(q, "IBE_OH_ORDER_
TOTAL"));
```

Example Date Formatting: IBE_OS_ORDER_DATE is an AK region item that corresponds to the Date column, which requires date formatting. You can invoke the addFormatter method as follows (assume q as an instantiated object of AkQuery):

```
q.addFormatter(AkDateFormatter.getAkDateFormatter(q, "IBE_OS_ORDER_DATE"));
```

connect

```
public final void connect(String name)
throws FrameworkException
```

Connects to the database and initializes the named query. Classes extending the Query abstract class based on AK regions should interpret the name parameter as the name of an AK region; classes based directly on database views should interpret name as the name of a database view. Names are case-sensitive. The programmer must ensure that queries initialized by calling connect() are properly cleaned up by calling disconnect() when finished.

Parameters: name - name of the query, e.g. AK region name or view name

Throws: FrameworkException if an error occurs while connecting to the database and/or initializing the query

Overrides: connect in class Query

See Also: disconnect

Example IBE_ORD_SUM_R is the AK region for displaying the Order Summary data. To initialize this region and retrieve all of the data related to this region, you can invoke the connect method as follows (assume q as an instantiated object of AkQuery):

```
q.connect("IBE_ORD_SUM_R");
```

disconnect

```
public final void disconnect()
```

Disconnects from the database and cleans up the query. The programmer must ensure that queries initialized by calling connect() are properly cleaned up by calling disconnect() when finished.

Overrides: disconnect in class Query

See Also: connect

execute

```
public final void execute()
throws FrameworkException, SQLException
```

Executes the query.

Throws: FrameworkException if an error occurred while fetching and/or formatting the query results

Overrides: execute in class Query

fromURL

```
public final void fromURL(ServletRequest request)
throws FrameworkException, QueryValidatorException
```

Reconstructs the state of a query from URL parameters generated by an HTML form or a call to toURL(). This method must be called after calling connect().

Parameters: request - the HTTP request containing the URL parameters

Throws: FrameworkException if the data in the URL parameters is corrupt and/or some error occurred while attempting to initialize the query object

Overrides: fromURL in class Query

getBatchSize

```
public final int getBatchSize()
```

Returns the maximum number of records to be displayed on a single page. If this method is invoked before connect(), it returns 0.

Returns: maximum number of rows per page

Overrides: getBatchSize in class Query

See Also: connect

getColumnIndex

```
public final int getColumnIndex(String itemName)
throws FrameworkException
```

Returns the index of the database view column on which the named query item is based. Item indexes are zero-based; item names are case- sensitive. If this method is invoked before connect(), the result is undefined.

Parameters: itemName - name of query item

Returns: database view column index of query item

Throws: FrameworkException if no item with the specified name is found

Overrides: getColumnIndex in class Query

See Also: connect

getConditions

```
public final Vector getConditions()
```

Returns a vector of all QueryCondition objects added to the query via the addCondition() methods. Used by classes implementing the QueryValidator interface.

Returns: vector containing all QueryConditions for this query

Overrides: getConditions in class Query

See Also: addCondition, addCondition, QueryValidator

getDateFormat

```
public final String getDateFormat()
```

Returns the Oracle date format mask (e.g. DD-MON-YYYY) used by the query. If this method is invoked before connect(), the result is undefined.

Returns: date format mask used by the query

Overrides: getDateFormat in class Query

See Also: connect

getInvoiceId

```
public String getInvoiceId(String lineId)
throws FrameworkException, SQLException
```

getItemIndex

```
public final int getItemIndex(String itemName)
throws FrameworkException
```

Returns the index of the query item (i.e. AK region item or view column) with the specified name. Item indexes are zero-based; item names are case-sensitive. If this method is invoked before connect(), the result is undefined.

Parameters: itemName - name of query item

Returns: index of query item

Throws: FrameworkException if no item with the specified name is found

Overrides: getItemIndex in class Query

See Also: getItemName, connect

getItemLabel

```
public final String getItemLabel(int itemIndex)
throws FrameworkException
```

Returns the display label of the query item with the specified index, suitable for use as a column heading when rendering the query result table on the page. Item indexes are zero-based. If this method is invoked before connect(), the result is undefined.

Parameters: itemIndex - index of query item

Returns: item label (i.e. column heading)

Throws: FrameworkException if itemIndex is invalid

Overrides: getItemLabel in class Query

See Also: getItemLabel, connect

getItemLabel

```
public final String getItemLabel(String itemName)
```

```
throws FrameworkException
```

Returns the display label of the query item with the specified name, suitable for use as a column heading when rendering the query result table on the page. Item names are case-sensitive. If this method is invoked before connect(), the result is undefined.

Parameters: itemName - name of query item

Returns: item label (i.e. column heading)

Throws: FrameworkException if no item with the specified name is found

Overrides: getItemLabel in class Query

See Also: getItemLabel, connect

getItemName

```
public final String getItemName(int itemIndex)
throws FrameworkException
```

Returns the name of the query item (i.e. AK region item or view column) with the specified index. Item indexes are zero-based; item names are case-sensitive. If this method is invoked before connect(), the result is undefined.

Parameters: itemIndex - index of query item

Returns: name of query item

Throws: FrameworkException if itemIndex is invalid

Overrides: getItemName in class Query

See Also: getItemIndex, connect

getName

```
public final String getName()
```

Returns a string that uniquely identifies the region (along with the responsibility ID, application ID, and language code used when connecting to the region). Used for caching purposes.

Returns: string uniquely identifying the region

Overrides: getName in class Query

See Also: connect

getNumItems

```
public final int getNumItems()
```

Returns the number of items (i.e. fields) in a record returned by the query. Classes extending the Query abstract class based on AK regions should return the number of region items; classes based directly on database views should return the number of view columns. If this method is invoked before connect(), it returns 0.

Returns: number of query items

Overrides: getNumItems in class Query

getNumRowsFetched

```
public final int getNumRowsFetched()
```

Returns the total number of rows fetched by the last call to execute(). Depending on the way the query statement is constructed, this value may be greater than or equal to the value returned by getNumRowsShown(). If this method is invoked before execute(), it returns 0.

Returns: total number of rows fetched

Overrides: getNumRowsFetched in class Query

See Also: getNumRowsShown, execute

getNumRowsShown

```
public final int getNumRowsShown()
```

Returns the actual number of records to be displayed on the current page. This number is always guaranteed to be less than or equal to the value returned by getBatchSize(). If this method is invoked before execute(), it returns 0.

Returns: number of rows shown on current page

Overrides: getNumRowsShown in class Query

See Also: getBatchSize, execute

getStartRow

```
public final int getStartRow()
```

Returns the index of the first row to be displayed on the current page. Row indexes are zero-based, so the index of the first row on the page displaying records 11-20 is actually 10. If this method is invoked before connect(), it returns 0.

Overrides: getStartRow in class Query

See Also: connect

getValue

```
public final String getValue(int rowIndex, int itemIndex)
throws FrameworkException
```

Returns the string representation of the value in the specified cell of the query result table. rowIndex is zero-based and must be strictly less than the value returned by getNumRowsShown(). itemIndex is zero-based and must be strictly less than the value returned by getNumItems(). Any formatting performed by QueryFormatters registered with the query will have already taken place and will be reflected in the value returned by getValue(). If this method is invoked before execute(), the result is undefined.

Parameters: rowIndex - row index of the cell to be displayed

itemIndex - column index of the cell to be displayed

Returns: value of the specified table cell

Throws: FrameworkException if either index is invalid

Overrides: getValue in class Query

See Also: getValue, getNumRowsShown, getNumItems, execute

getValue

```
public final String getValue(int rowIndex, String itemName)
throws FrameworkException
```

Returns the string representation of the value in the specified cell of the query result table. rowIndex is zero-based and must be strictly less than the value returned by getNumRowsShown(). itemName is case-sensitive and must be one of the item names returned by getItemName(). Any formatting performed by QueryFormatters registered with the query will have already taken place and will be reflected in the value returned by getValue(). If this method is invoked before execute(), the result is undefined.

Parameters: rowIndex - row index of the cell to be displayed itemName - item name of the cell to be displayed

Returns: value of the specified table cell

Throws: FrameworkException if rowIndex is invalid and/or itemName is not found

Overrides: getValue in class Query

See Also: getValue, getNumRowsShown, getItemName, execute

isDisplayable

```
public final boolean isDisplayable(int itemIndex)
throws FrameworkException
```

Returns true if the specified query item may be displayed. This is purely a convenience feature; the displaying JSP may decide whether or not to check if a particular item is displayable before calling getValue(). Item indexes are zero-based. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - index of query item

Returns: true if the item should be displayed

Throws: FrameworkException if itemIndex is invalid

Overrides: isDisplayable in class Query

See Also: isDisplayable, connect

isDisplayable

```
public final boolean isDisplayable(String itemName)
throws FrameworkException
```

Returns true if the specified query item may be displayed. This is purely a convenience feature; the displaying JSP may decide whether or not to check if a particular item is displayable before calling getValue(). Item names are case-sensitive. If this method is called before connect(), the result is undefined.

Parameters: itemName - name of query item

Returns: true if the item should be displayed

Throws: FrameworkException if itemName is not found

Overrides: isDisplayable in class Query

See Also: isDisplayable, connect

isQueryable

```
public final boolean isQueryable(int itemIndex)
throws FrameworkException
```

Returns true if the specified query item may be searched on. Item indexes are zero-based. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - index of query item

Returns: true if the item can be searched on

Throws: FrameworkException if itemIndex is invalid

Overrides: isQueryable in class Query

See Also: isQueryable, connect

isQueryable

```
public final boolean isQueryable(String itemName)
throws FrameworkException
```

Returns true if the specified query item may be searched on. Item names are case-sensitive. If this method is called before connect(), the result is undefined.

Parameters: itemName - name of query item

Returns: true if the item can be searched on

Throws: FrameworkException

if itemName is not found

Overrides: isQueryable in class Query

See Also: isQueryable, connect

isSelectable

```
public final boolean isSelectable(int itemIndex)
throws FrameworkException
```

Returns true if the specified query item is based on a database object and thus may be included in a SELECT or ORDER BY clause. Classes extending the Query abstract class based on AK regions should return true if the specified AK region item is based on an object attribute; classes based directly on database views should return true if the specified query item is based on a view column. In any case, all queryable items must also be selectable, though the reverse is not necessarily true.

Item indexes are zero-based. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - index of query item

Returns: true if the item is based on a database object

Throws: FrameworkException if itemIndex is invalid

Overrides: isSelectable in class Query

See Also: isSelectable, isQueryable, connect

isSelectable

```
public final boolean isSelectable(String itemName)
throws FrameworkException
```

Returns true if the specified query item is based on a database object and thus may be included in a SELECT or ORDER BY clause. Classes extending the query abstract class based on AK regions should return true if the specified AK region item is based on an object attribute; classes based directly on database views should return true if the specified query item is based on a view column. In any case, all queryable items must also be selectable, though the reverse is not necessarily true. Item names are case-sensitive. If this method is called before connect(), the result is undefined.

Parameters: itemName - name of query item

Returns: true if the item is based on a database object

Throws: FrameworkException if itemName is not found

Overrides: isSelectable in class Query

See Also: isSelectable, isQueryable, connect

resetConditions

```
public final void resetConditions()
```

Removes all conditions from the WHERE clause of the query. If this method is called before connect(), the result is undefined.

Overrides: resetConditions in class Query

See Also: addCondition, addCondition, connect

setBatchSize

```
public final void setBatchSize(int batchSize)
```

```
throws FrameworkException
```

Sets the batch size (i.e. the maximum number of records shown per page) to the specified value. batchSize must be a non-negative integer.

Parameters: batchSize - new number of rows per page

Throws: FrameworkException if batchSize is not a non-negative integer

Overrides: setBatchSize in class Query

setOrderByColumn

```
public final void setOrderByColumn(int itemIndex, boolean isAscending)
throws FrameworkException
```

Sets the ORDER BY clause of the query to use the specified item and sort direction. The item must be one for which isSelectable() returns true. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - index of query item

isAscending - true if the sort direction should be ascending, false if the sort direction should be descending

Throws: FrameworkException if itemIndex is invalid

Overrides: setOrderByColumn in class Query

See Also: setOrderByColumn, isSelectable, connect

setOrderByColumn

```
public final void setOrderByColumn(String itemName, boolean isAscending)
throws FrameworkException
```

Sets the ORDER BY clause of the query to use the specified item and sort direction. The item must be one for which isSelectable() returns true. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - name of query item

isAscending - true if the sort direction should be ascending, false if the sort direction should be descending

Throws: FrameworkException if itemName is not found

Overrides: setOrderByColumn in class Query

See Also: setOrderByColumn, isSelectable, connect

Example In the Order Summary page, if the query should have an Order By with Order Number in Descending direction, you can invoke the setOrderByColumn method as follows (assume q as an instantiated object of AkQuery):

```
q.setOrderByColumn("IBE_OS_ORDER_NUMBER",false);
```

where IBE_OS_ORDER_NUMBER is an AK region item for the Order_number column.

setOrderByColumn

```
public final void setOrderByColumn(int itemIndex)
throws FrameworkException
```

Sets the ORDER BY clause of the query to use the specified item. If the ORDER BY clause is already using this item, reverses the sort direction. If the ORDER BY clause is empty or is using a different item, sets the ORDER BY clause to use this item and sets the sort direction to ascending by default. The item must be one for which isSelectable() returns true. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - index of query item

Throws: FrameworkException if itemIndex is invalid

Overrides: setOrderByColumn in class Query

See Also: setOrderByColumn, isSelectable, connect

setOrderByColumn

```
public final void setOrderByColumn(String itemName)
throws FrameworkException
```

Sets the ORDER BY clause of the query to use the specified item. If the ORDER BY clause is already using this item, reverses the sort direction. If the ORDER BY clause is empty or is using a different item, sets the ORDER BY clause to use this item and sets the sort direction to ascending by default. The item must be one for which isSelectable() returns true. If this method is called before connect(), the result is undefined.

Parameters: itemName - name of query item

Throws: FrameworkException if itemName is not found

Overrides: setOrderByColumn in class Query

See Also: setOrderByColumn, isSelectable, connect

setStartRow

```
public final void setStartRow(int rowIndex)
throws FrameworkException
```

Sets the start row to the specified row for execution. rowIndex must be a non-negative integer.

Parameters: rowIndex - index of the start row

Throws: FrameworkException if rowIndex is not a non-negative integer

Overrides: setStartRow in class Query

showQueryConditions

```
public final String showQueryConditions()
```

Returns a properly formatted HTML option list of search conditions. Each query item for which isQueryable() returns true represents a valid search condition. The return value is a string consisting of OPTION tags, suitable to be included between a SELECT tag and its closing tag. If preset is non-NULL and specifies one of the values in the list, the corresponding option will be automatically selected. If this method is called before connect(), the result is undefined.

Parameters: preset - preset value for the option list

Returns: a list of OPTION tags specifying valid query criteria

Overrides: showQueryConditions in class Query

See Also: isQueryable, connect

showQueryOperators

```
public final String showQueryOperators()
throws FrameworkException
```

Returns a properly formatted HTML option list of search operators. The return value is a string consisting of OPTION tags, suitable to be included between a SELECT tag and its closing tag. If preset is non-NULL and specifies one of the values in the list, the corresponding option will be automatically selected. For queries based on AK regions, the valid operators are AIS, BNOT, CCONTAIN, DSTART, EEND, FGREATER, and GLESS. If this method is called before connect(), the result is undefined.

Parameters: preset - preset value for the option list

Returns: a list of OPTION tags specifying valid query operators

Overrides: showQueryOperators in class Query

See Also: connect

toURL

```
public final String toURL()
```

Externalizes the current state of the query as a URL parameter string of the form "param1=val1¶m2=val2&...¶mX=valX". When creating hyperlinks from one query display page to another, programmers must ensure that the string returned by toURL() is appended to the URL parameters of the link.

Overrides: toURL in class Query

5.5 Class AkQueryCondition

```
java.lang.Object > oracle.apps.ibe.postsales.QueryCondition >
oracle.apps.ibe.postsales.AkQueryCondition
```

public final class **AkQueryCondition**

extends QueryCondition

AkQueryCondition encapsulates a single condition in the WHERE clause of an @see AkQuery. The WHERE clause of an AkQuery is a conjunction of zero or more AkQueryConditions. Note that AkQueryCondition does not support disjunctions of conditions or joins between database objects.

See Also: AkQuery, Region, AkRegionItem

5.5.1 Variables for Class AkQueryCondition

_itemIndex

```
protected int _itemIndex
```

_operatorCode

```
protected String _operatorCode
```

RCS_ID

```
public static final String RCS_ID
```

RCS_ID_RECORDED

```
public static final boolean RCS_ID_RECORDED
```

thisClass

```
public static final String thisClass
```

_value

```
protected String _value
```

5.5.2 Constructors for Class AkQueryCondition

AkQueryCondition

```
public AkQueryCondition(int itemIndex, String operatorCode, String value)
```

Constructs a AkQueryCondition instance and initializes it with the values provided.

Parameters: itemIndex - zero-based index of the query item

operatorCode - SQL binary operator

value - value to compare against

5.5.3 Methods for Class AkQueryCondition

The following table is an index of Class AkQueryCondition methods:

Table 5–5 Method Index for Class AkQueryCondition

Method	Description
getItemIndex	Returns the item index
getOperatorCode	Returns the operator code
getValue	Returns the value

getItemIndex

```
public final int getItemIndex()
```

Returns the item index

Overrides: getItemIndex in class QueryCondition

getOperatorCode

```
public final String getOperatorCode()
```

Returns the operator code

Overrides: getOperatorCode in class QueryCondition

getValue

```
public final String getValue()
```

Returns the value

Overrides: getValue in class QueryCondition

5.6 Class AkRegion

```
java.lang.Object > oracle.apps.ibe.postsales.AkRegion
```

public final class **AkRegion**

extends Object

The AkRegion class encapsulates AK regions for post-sales (e.g. Order Tracker) queries.

5.6.1 Variables for Class AkRegion

RCS_ID

```
public static final String RCS_ID
```

RCS_ID_RECORDED

```
public static final boolean RCS_ID_RECORDED
```

thisClass

```
public static final String thisClass
```

5.6.2 Methods for Class AkRegion

The following table is an index of Class AkRegion methods:

Table 5–6 Method Index for Class AkRegion

Method	Description
getAkRegion	Returns an AkRegion object instance based on the region name, the responsibility ID, the application ID, and the language code
getColumnIndex	
getColumnName	
getColumnType	
getItemIndex	Returns the index of the AK region item with the specified name
getItemLabel	Returns the display label of the region item with the specified index, suitable for use as a column heading when rendering the query result table on the page
getItemName	Returns the name of the AK region item with the specified index
getNumItems	Returns the total number of region items in the AK region
getRegionName	Returns the name of the AK region
getViewName	Returns the name of the view on which the AK region is based
isDisplayable	Returns true if the specified query item may be displayed
isQueryable	Returns true if the specified query item may be searched on
isSelectable	Returns true if the specified query item is based on a database object and thus may be included in a SELECT or ORDER BY clause
showQueryConditions	Returns a properly formatted HTML option list of search conditions based on queryable AK region items
showQueryOperators	Returns a properly formatted HTML option list of search operators

getAkRegion

```
public static final AkRegion getAkRegion(OracleConnection conn, String
regionName, int respId, int appId, String langCode)
throws SQLException, FrameworkException
```

Returns an AkRegion object instance based on the region name, the responsibility ID, the application ID, and the language code

getColumnIndex

```
public final int getColumnIndex(int itemIndex)
throws FrameworkException
```

getColumnIndex

```
public final int getColumnIndex(String itemName)
throws FrameworkException
```

getColumnName

```
public final String getColumnName(int itemIndex)
throws FrameworkException
```

getColumnName

```
public final String getColumnName(String itemName)
throws FrameworkException
```

getColumnType

```
public final String getColumnType(int itemIndex)
throws FrameworkException
```

getColumnType

```
public final String getColumnType(String itemName)
throws FrameworkException
```

getItemIndex

```
public final int getItemIndex(String itemName)
throws FrameworkException
```

Returns the index of the AK region item with the specified name. Item indexes are zero-based; item names are case-sensitive.

Parameters: itemName - name of AK region item

Returns: index of AK region item

Throws: FrameworkException if no item with the specified name is found

See Also: getItemName

getItemLabel

```
public final String getItemLabel(int itemIndex)
throws FrameworkException
```

Returns the display label of the region item with the specified index, suitable for use as a column heading when rendering the query result table on the page. Item indexes are zero-based.

Parameters: itemIndex - index of region item

Returns: item label (i.e. column heading)

Throws: FrameworkException if itemIndex is invalid

See Also: getItemLabel, connect

getItemLabel

```
public final String getItemLabel(String itemName)
throws FrameworkException
```

Returns the display label of the region item with the specified name, suitable for use as a column heading when rendering the query result table on the page. Item names are case-sensitive.

Parameters: itemName - name of region item

Returns: item label (i.e. column heading)

Throws: FrameworkException if no item with the specified name is found

See Also: getItemLabel, connect

getItemName

```
public final String getItemName(int itemIndex)
throws FrameworkException
```

Returns the name of the AK region item with the specified index. Item indexes are zero-based; item names are case-sensitive.

Parameters: itemIndex - index of AK region item

Returns: name of AK region item

Throws: FrameworkException if itemIndex is invalid

See Also: getItemIndex

getNumItems

```
public final int getNumItems()
```

Returns the total number of region items in the AK region

Returns: number of AK region items

getRegionName

```
public final String getRegionName()
```

Returns the name of the AK region

Returns: name of AK region

getViewName

```
public final String getViewName()
```

Returns the name of the view on which the AK region is based

Returns: name of database view

isDisplayable

```
public final boolean isDisplayable(int itemIndex)
throws FrameworkException
```

Returns true if the specified query item may be displayed. This is purely a convenience feature; the displaying JSP may decide whether or not to check if a particular item is displayable before calling getValue(). Item indexes are zero-based. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - index of query item

Returns: true if the item should be displayed

Throws: FrameworkException if itemIndex is invalid

See Also: isDisplayable, connect

isDisplayable

```
public final boolean isDisplayable(String itemName)
throws FrameworkException
```

Returns true if the specified query item may be displayed. This is purely a convenience feature; the displaying JSP may decide whether or not to check if a particular item is displayable before calling getValue(). Item names are case-sensitive. If this method is called before connect(), the result is undefined.

Parameters: itemName - name of query item

Returns: true if the item should be displayed

Throws: FrameworkException if itemName is not found

See Also: isDisplayable, connect

isQueryable

```
public final boolean isQueryable(int itemIndex)
throws FrameworkException
```

Returns true if the specified query item may be searched on. Item indexes are zero-based. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - index of query item

Returns: true if the item can be searched on

Throws: FrameworkException if itemIndex is invalid

See Also: isQueryable, connect

isQueryable

```
public final boolean isQueryable(String itemName)
throws FrameworkException
```

Returns true if the specified query item may be searched on. Item names are case-sensitive. If this method is called before connect(), the result is undefined.

Parameters: itemName - name of query item

Returns: true if the item can be searched on

Throws: FrameworkException if itemName is not found

See Also: isQueryable, connect

isSelectable

```
public final boolean isSelectable(int itemIndex)
throws FrameworkException
```

Returns true if the specified query item is based on a database object and thus may be included in a SELECT or ORDER BY clause. Classes extending the Query abstract class based on AK regions should return true if the specified AK region item is based on an object attribute; classes based directly on database views should return true if the specified query item is based on a view column. In any case, all queryable items must also be selectable, though the reverse is not necessarily true. Item indexes are zero-based. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - index of query item

Returns: true if the item is based on a database object

Throws: FrameworkException if itemIndex is invalid

See Also: isSelectable, isQueryable, connect

isSelectable

```
public final boolean isSelectable(String itemName)
throws FrameworkException
```

Returns true if the specified query item is based on a database object and thus may be included in a SELECT or ORDER BY clause. Classes extending the query abstract class based on AK regions should return true if the specified AK region item is based on an object attribute; classes based directly on database views should return true if the specified query item is based on a view column. In any case, all queryable items must also be selectable, though the reverse is not necessarily true. Item names are case-sensitive. If this method is called before connect(), the result is undefined.

Parameters: itemName - name of query item

Returns: true if the item is based on a database object

Throws: FrameworkException if itemName is not found

See Also: isSelectable, isQueryable, connect

showQueryConditions

```
public final String showQueryConditions()
```

Returns a properly formatted HTML option list of search conditions based on queryable AK region items. Each queryable region item is assumed to represent a valid search condition. The return value is a string consisting of OPTION tags, suitable to be included between a SELECT tag and its closing tag. If preset is non-NULL and specifies one of the values in the list, the corresponding option will be automatically selected.

Parameters: preset - preset value for the option list

Returns: a list of OPTION tags specifying valid query criteria

See Also: connect

showQueryOperators

```
public final String showQueryOperators()
```

Returns a properly formatted HTML option list of search operators. The return value is a string consisting of OPTION tags, suitable to be included between a SELECT tag and its closing tag. If preset is non-NULL and specifies one of the values in the list, the corresponding option will be automatically selected. For queries based on AK regions, the valid operators are AIS, BNOT, CCONTAIN, DSTART, EEND, FGREATER, and GLESS. If this method is called before connect(), the result is undefined.

Returns: a list of OPTION tags specifying valid query operators

5.7 Class IbeAtpPvt

```
java.lang.Object > oracle.apps.ibe.postsales.IbeAtpPvt
```

public class **IbeAtpPvt**

extends Object

IbeAtpPvt is a Rosetta-generated Java wrapper for the IBE_ATP_PVT PL/SQL package. It contains a single static class (AtpLineTyp), which is a wrapper for the PL/SQL type IBE_ATP_PVT.Atp_Line_Typ, as well as a static method (CheckAvailability), which is a wrapper to call the PL/SQL procedure IBE_ATP_PVT.Check_Availability.

5.7.1 Variables for Class IbeAtpPvt

RCS_ID

```
public static final String RCS_ID
```

5.7.2 Constructors for Class IbeAtpPvt

IbeAtpPvt

```
public IbeAtpPvt()
```

5.7.3 Methods for Class IbeAtpPvt

The following table is an index of Class IbeAtpPvt methods:

Table 5–7 Method Index for Class IbeAtpPvt

Method	Description
checkAvailability	This is a Rosetta-generated method to call the PL/SQL procedure IBE_ATP_PVT.Check_Availability
writeOrSkip	

checkAvailability

```
public static void checkAvailability(OracleConnection _connection, BigDecimal p_
quote_header_id, String p_date_format, String p_lang_code, String x_error_
flag[], String x_error_message[], IbeAtpPvt. AtpLineTyp x_atp_line_tbl[][])
throws SQLException
```

This is a Rosetta-generated method to call the PL/SQL procedure IBE_ATP_PVT.Check_Availability. The parameter x_atp_line_tbl is a nested array of IbeAtpPvt.AtpLineTyp objects. It must be declared by the caller as IbeAtpPvt.AtpLineTyp[][] x_atp_line_tbl = new IbeAtpPvt.AtpLineTyp[1][]. For each line, the following fields must be populated: quote_line_id, organization_id, inventory_item_id, quantity, and uom_code. The request_date parameter is optional; if null or the empty string, the API assumes SYSDATE. The checkAvailability() method populates the following fields for each quote line: request_date_quantity and available_date.

Parameters: _connection - database connection--IN

p_quote_header_id - shopping cart ID--IN

p_date_format - SQL date format string--IN

p_lang_code - user’s language--IN

x_error_flag - Y if an error occurred, N otherwise--OUT

x_error_message - description of the error--OUT x_atp_line_tbl - array of quote lines--IN/OUT

Throws: SQLException if a database or SQL error occurs

writeOrSkip

```
protected static boolean writeOrSkip(StringBuffer sb, boolean isGMiss, boolean
anyWritten, int numSkipped, String argName)
```

5.8 Class Query

```
java.lang.Object > oracle.apps.ibe.postsales.Query
```

public abstract class **Query**

extends Object

The Query abstract class specifies the minimum functionality required for database queries on which the iStore post-sales pages (e.g. Order Tracker) depend.

See Also: QueryFormatter, QueryValidator, QueryUtil

5.8.1 Variables for Class Query

RCS_ID

```
public static final String RCS_ID
```

RCS_ID_RECORDED

```
public static final boolean RCS_ID_RECORDED
```

5.8.2 Constructors for Class Query

Query

```
public Query()
```

5.8.3 Methods for Class Query

The following table is an index of Class Query methods:

Table 5–8 Method Index for Class Query

Method	Description
addCondition	Adds a condition to the WHERE clause of the query
addFormatter	Registers a QueryFormatter object for the specified query item
connect	Connects to the database and initializes the named query
disconnect	Disconnects from the database and cleans up the query
execute	Executes the query
fromURL	Reconstructs the state of a query from URL parameters generated by an HTML form or a call to toURL()
getBatchSize	Returns the maximum number of records to be displayed on a single page
getColumnIndex	Returns the index of the database view column on which the named query item is based
getConditions	Returns a vector of all QueryCondition objects added to the query via the addCondition() methods
getDateFormat	Returns the Oracle date format mask (e.g. DD-MON-YYYY) used by the query
getItemIndex	Returns the index of the query item (i.e. AK region item or view column) with the specified name
getItemLabel	Returns the display label of the query item with the specified index, suitable for use as a column heading when rendering the query result table on the page
getItemName	Returns the name of the query item (i.e. AK region item or view column) with the specified index
getName	Returns the name of the query
getNumItems	Returns the number of items (i.e. fields) in a record returned by the query
getNumRowsFetched	Returns the total number of rows fetched by the last call to execute()
getNumRowsShown	Returns the actual number of records to be displayed on the current page
getStartRow	Returns the index of the first row to be displayed on the current page
getValue	Returns the string representation of the value in the specified cell of the query result table

Table 5–8 Method Index for Class Query (Cont.)

Method	Description
isDisplayable	Returns true if the specified query item may be displayed
isQueryable	Returns true if the specified query item may be searched on
isSelectable	Returns true if the specified query item is based on a database object and thus may be included in a SELECT or ORDER BY clause
resetConditions	Removes all conditions from the WHERE clause of the query
setBatchSize	Sets the batch size (i.e. the maximum number of records shown per page) to the specified value
setOrderByColumn	Sets the ORDER BY clause of the query to use the specified item and sort direction
setStartRow	Sets the start row to the specified row for execution
showQueryConditions	Returns a properly formatted HTML option list of search conditions
showQueryOperators	Returns a properly formatted HTML option list of search operators
toURL	Externalizes the current state of the query as a URL parameter string of the form "param1=val1¶m2=val2&...¶mX=valX"

addCondition

```
public abstract void addCondition(int itemIndex, String operatorCode, String
value)
throws FrameworkException
```

Adds a condition to the WHERE clause of the query. Successive calls to addCondition() add new conditions to the WHERE clause in order. The condition is of the form CUST_ACCOUNT_ID = 1001. The specified item must be one for which isSelectable() returns true. The specified operator must be one of the valid operators returned by showQueryOperators(). For queries based on AK regions, valid operators are AIS, BNOT, CCONTAIN, DSTART, EEND, FGREATER, and GLESS. The value may be any free-form String, but it must be convertible to the specified item's datatype. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - index of query item

operatorCode - valid binary operator code

value - value against which item is to be compared

Throws: FrameworkException if itemIndex is invalid

See Also: addCondition, isSelectable, showQueryOperators, connect

addCondition

```
public abstract void addCondition(String itemName, String operatorCode, String
value)
throws FrameworkException
```

Adds a condition to the WHERE clause of the query. Successive calls to addCondition() add new conditions to the WHERE clause in order. The condition is of the form CUST_ACCOUNT_ID = 1001. The specified item must be one for which isSelectable() returns true. The specified operator must be one of the valid operators returned by showQueryOperators(). For queries based on AK regions, valid operators are AIS, BNOT, CCONTAIN, DSTART, EEND, FGREATER, and GLESS. The value may be any free-form String, but it must be convertible to the specified item's datatype. If this method is called before connect(), the result is undefined.

Parameters: itemName - name of query item

operatorCode - valid binary operator code

value - value against which item is to be compared

Throws: FrameworkException if itemName is not found

See Also: addCondition, isSelectable, showQueryOperators, connect

addCondition

```
public abstract void addCondition(String condition)
```

Appends the specified string to the WHERE clause of the query as is, without performing any syntactic or semantic verification. The programmer must ensure that the argument is a valid condition given the specifics of the query. If this method is called before connect(), the result is undefined.

Parameters: condition - any valid condition to be appended to the WHERE clause

See Also: addCondition, addCondition, connect

addFormatter

```
public abstract void addFormatter(QueryFormatter formatter)
throws FrameworkException
```

Registers a QueryFormatter object for the specified query item. As it fetches data from the database, the execute() method invokes the appropriate QueryFormatter objects to format specific query items in the desired way.

Parameters: formatter - the QueryFormatter object to be registered

itemIndex - index of the query item to be formatted

connect

```
public abstract void connect(String name)
throws FrameworkException
```

Connects to the database and initializes the named query. Classes extending the Query abstract class based on AK regions should interpret the name parameter as the name of an AK region; classes based directly on database views should interpret name as the name of a database view. Names are case-sensitive. The programmer must ensure that queries initialized by calling connect() are properly cleaned up by calling disconnect() when finished.

Parameters: name - name of the query, e.g. AK region name or view name

Throws: FrameworkException if an error occurs while connecting to the database and/or initializing the query

See Also: disconnect

disconnect

```
public abstract void disconnect()
```

Disconnects from the database and cleans up the query. The programmer must ensure that queries initialized by calling connect() are properly cleaned up by calling disconnect() when finished.

See Also: connect

execute

```
public abstract void execute()
throws FrameworkException, SQLException
```

Executes the query

Throws: FrameworkException if an error occurred while fetching and/or formatting the query results

fromURL

```
public abstract void fromURL(ServletRequest request)
throws FrameworkException
```

Reconstructs the state of a query from URL parameters generated by an HTML form or a call to toURL(). This method must be called after calling connect().

Parameters: request - the HTTP request containing the URL parameters

Throws: FrameworkException if the data in the URL parameters is corrupt and/or some error occurred while attempting to initialize the query object

getBatchSize

```
public abstract int getBatchSize()
```

Returns the maximum number of records to be displayed on a single page. If this method is invoked before connect(), it returns 0.

Returns: maximum number of rows per page

See Also: connect

getColumnIndex

```
public abstract int getColumnIndex(String itemName)
throws FrameworkException
```

Returns the index of the database view column on which the named query item is based. Item indexes are zero-based; item names are case- sensitive. If this method is invoked before connect(), the result is undefined.

Parameters: itemName - name of query item

Returns: database view column index of query item

Throws: FrameworkException if no item with the specified name is found

See Also: connect

getConditions

```
public abstract Vector getConditions()
```

Returns a vector of all QueryCondition objects added to the query via the addCondition() methods. Used by classes implementing the QueryValidator interface.

Returns: vector containing all QueryConditions for this query

See Also: addCondition, addCondition, QueryValidator

getDateFormat

```
public abstract String getDateFormat()
```

Returns the Oracle date format mask (e.g. DD-MON-YYYY) used by the query. If this method is invoked before connect(), the result is undefined.

Returns: date format mask used by the query

See Also: connect

getItemIndex

```
public abstract int getItemIndex(String itemName)
throws FrameworkException
```

Returns the index of the query item (i.e. AK region item or view column) with the specified name. Item indexes are zero-based; item names are case-sensitive. If this method is invoked before connect(), the result is undefined.

Parameters: itemName - name of query item

Returns: index of query item

Throws: FrameworkException if no item with the specified name is found

See Also: getItemName, connect

getItemLabel

```
public abstract String getItemLabel(int itemIndex)
throws FrameworkException
```

Returns the display label of the query item with the specified index, suitable for use as a column heading when rendering the query result table on the page. Item indexes are zero-based. If this method is invoked before connect(), the result is undefined.

Parameters: itemIndex - index of query item

Returns: item label (i.e. column heading)

Throws: FrameworkException if itemIndex is invalid

See Also: getItemLabel, connect

getItemLabel

```
public abstract String getItemLabel(String itemName)
throws FrameworkException
```

Returns the display label of the query item with the specified name, suitable for use as a column heading when rendering the query result table on the page. Item names are case-sensitive. If this method is invoked before connect(), the result is undefined.

Parameters: itemName - name of query item

Returns: item label (i.e. column heading)

Throws: FrameworkException if no item with the specified name is found

See Also: getItemLabel, connect

getItemName

```
public abstract String getItemName(int itemIndex)
throws FrameworkException
```

Returns the name of the query item (i.e. AK region item or view column) with the specified index. Item indexes are zero-based; item names are case-sensitive. If this method is invoked before connect(), the result is undefined.

Parameters: itemIndex - index of query item

Returns: name of query item

Throws: FrameworkException if itemIndex is invalid

See Also: getItemIndex, connect

getName

```
public abstract String getName()
```

Returns the name of the query. Classes extending the Query abstract class based on AK regions should return a string that uniquely identifies the region; classes based directly on database views should return a string that uniquely identifies the view. If this method is invoked before connect(), the result is undefined.

Returns: name of the query

See Also: connect

getNumItems

```
public abstract int getNumItems()
```

Returns the number of items (i.e. fields) in a record returned by the query. Classes extending the Query abstract class based on AK regions should return the number of region items; classes based directly on database views should return the number of view columns. If this method is invoked before connect(), it returns 0.

Returns: number of query items

getNumRowsFetched

```
public abstract int getNumRowsFetched()
```

Returns the total number of rows fetched by the last call to execute(). Depending on the way the query statement is constructed, this value may be greater than or equal to the value returned by getNumRowsShown(). If this method is invoked before execute(), it returns 0.

Returns: total number of rows fetched

See Also: getNumRowsShown, execute

getNumRowsShown

```
public abstract int getNumRowsShown()
```

Returns the actual number of records to be displayed on the current page. This number is always guaranteed to be less than or equal to the value returned by getBatchSize(). If this method is invoked before execute(), it returns 0.

Returns: number of rows shown on current page

See Also: getBatchSize, execute

getStartRow

```
public abstract int getStartRow()
```

Returns the index of the first row to be displayed on the current page. Row indexes are zero-based, so the index of the first row on the page displaying records 11-20 is actually 10. If this method is invoked before connect(), it returns 0.

See Also: connect

getValue

```
public abstract String getValue(int rowIndex, int itemIndex)
throws FrameworkException
```

Returns the string representation of the value in the specified cell of the query result table. rowIndex is zero-based and must be strictly less than the value returned by getNumRowsShown(). itemIndex is zero-based and must be strictly less than the value returned by getNumItems(). Any formatting performed by QueryFormatters registered with the query will have already taken place and will be reflected in the value returned by getValue(). If this method is invoked before execute(), the result is undefined.

Parameters: rowIndex - row index of the cell to be displayed

itemIndex - column index of the cell to be displayed

Returns: value of the specified table cell

Throws: FrameworkException if either index is invalid

See Also: getValue, getNumRowsShown, getNumItems, execute

getValue

```
public abstract String getValue(int rowIndex, String itemName)
throws FrameworkException
```

Returns the string representation of the value in the specified cell of the query result table. rowIndex is zero-based and must be strictly less than the value returned by getNumRowsShown(). itemName is case-sensitive and must be one of the item names returned by getItemName(). Any formatting performed by QueryFormatters registered with the query will have already taken place and will be reflected in the value returned by getValue(). If this method is invoked before execute(), the result is undefined.

Parameters: rowIndex - row index of the cell to be displayed

itemName - item name of the cell to be displayed

Returns: value of the specified table cell

Throws: FrameworkException if rowIndex is invalid and/or itemName is not found

See Also: getValue, getNumRowsShown, getItemName, execute

isDisplayable

```
public abstract boolean isDisplayable(int itemIndex)
throws FrameworkException
```

Returns true if the specified query item may be displayed. This is purely a convenience feature; the displaying JSP may decide whether or not to check if a particular item is displayable before calling getValue(). Item indexes are zero-based. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - index of query item

Returns: true if the item should be displayed

Throws: FrameworkException if itemIndex is invalid

See Also: isDisplayable, connect

isDisplayable

```
public abstract boolean isDisplayable(String itemName)
throws FrameworkException
```

Returns true if the specified query item may be displayed. This is purely a convenience feature; the displaying JSP may decide whether or not to check if a particular item is displayable before calling getValue(). Item names are case-sensitive. If this method is called before connect(), the result is undefined.

Parameters: itemName - name of query item

Returns: true if the item should be displayed

Throws: FrameworkException if itemName is not found

See Also: isDisplayable, connect

isQueryable

```
public abstract boolean isQueryable(int itemIndex)
throws FrameworkException
```

Returns true if the specified query item may be searched on. Item indexes are zero-based. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - index of query item

Returns: true if the item can be searched on

Throws: FrameworkException if itemIndex is invalid

See Also: isQueryable, connect

isQueryable

```
public abstract boolean isQueryable(String itemName)
throws FrameworkException
```

Returns true if the specified query item may be searched on. Item names are case-sensitive. If this method is called before connect(), the result is undefined.

Parameters: itemName - name of query item

Returns: true if the item can be searched on

Throws: FrameworkException if itemName is not found

See Also: isQueryable, connect

isSelectable

```
public abstract boolean isSelectable(int itemIndex)
throws FrameworkException
```

Returns true if the specified query item is based on a database object and thus may be included in a SELECT or ORDER BY clause. Classes extending the Query abstract class based on AK regions should return true if the specified AK region item is based on an object attribute; classes based directly on database views should return true if the specified query item is based on a view column. In any case, all queryable items must also be selectable, though the reverse is not necessarily true. Item indexes are zero-based. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - index of query item

Returns: true if the item is based on a database object

Throws: FrameworkException if itemIndex is invalid

See Also: isSelectable, isQueryable, connect

isSelectable

```
public abstract boolean isSelectable(String itemName)
throws FrameworkException
```

Returns true if the specified query item is based on a database object and thus may be included in a SELECT or ORDER BY clause. Classes extending the query abstract class based on AK regions should return true if the specified AK region item is based on an object attribute; classes based directly on database views should return true if the specified query item is based on a view column. In any case, all queryable items must also be selectable, though the reverse is not necessarily true. Item names are case-sensitive. If this method is called before connect(), the result is undefined.

Parameters: itemName - name of query item

Returns: true if the item is based on a database object

Throws: FrameworkException if itemName is not found

See Also: isSelectable, isQueryable, connect

resetConditions

```
public abstract void resetConditions()
```

Removes all conditions from the WHERE clause of the query. If this method is called before connect(), the result is undefined.

See Also: addCondition, addCondition, connect

setBatchSize

```
public abstract void setBatchSize(int batchSize)
throws FrameworkException
```

Sets the batch size (i.e. the maximum number of records shown per page) to the specified value. batchSize must be a non-negative integer.

Parameters: batchSize - new number of rows per page

Throws: FrameworkException if batchSize is not a non-negative integer

setOrderByColumn

```
public abstract void setOrderByColumn(int itemIndex, boolean isAscending)
throws FrameworkException
```

Sets the ORDER BY clause of the query to use the specified item and sort direction. The item must be one for which isSelectable() returns true. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - index of query item

isAscending - true if the sort direction should be ascending

Throws: FrameworkException if itemIndex is invalid

See Also: setOrderByColumn, isSelectable, connect

setOrderByColumn

```
public abstract void setOrderByColumn(String itemName, boolean isAscending)
throws FrameworkException
```

Sets the ORDER BY clause of the query to use the specified item and sort direction. The item must be one for which isSelectable() returns true. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - name of query item

isAscending - true if the sort direction should be ascending

Throws: FrameworkException if itemName is not found

See Also: setOrderByColumn, isSelectable, connect

setOrderByColumn

```
public abstract void setOrderByColumn(int itemIndex)
throws FrameworkException
```

Sets the ORDER BY clause of the query to use the specified item. If the ORDER BY clause is already using this item, reverses the sort direction. If the ORDER BY clause is empty or is using a different item, sets the ORDER BY clause to use this item and sets the sort direction to ascending by default. The item must be one for which isSelectable() returns true. If this method is called before connect(), the result is undefined.

Parameters: itemIndex - index of query item

Throws: FrameworkException if itemIndex is invalid

See Also: setOrderByColumn, isSelectable, connect

setOrderByColumn

```
public abstract void setOrderByColumn(String itemName)
throws FrameworkException
```

Sets the ORDER BY clause of the query to use the specified item. If the ORDER BY clause is already using this item, reverses the sort direction. If the ORDER BY clause is empty or is using a different item, sets the ORDER BY clause to use this item and sets the sort direction to ascending by default. The item must be one for which isSelectable() returns true. If this method is called before connect(), the result is undefined.

Parameters: itemName - name of query item

Throws: FrameworkException if itemName is not found

See Also: setOrderByColumn, isSelectable, connect

setStartRow

```
public abstract void setStartRow(int rowIndex)
throws FrameworkException
```

Sets the start row to the specified row for execution. rowIndex must be a non-negative integer.

Parameters: rowIndex - index of the start row

Throws: FrameworkException if rowIndex is not a non-negative integer

showQueryConditions

```
public abstract String showQueryConditions()
```

Returns a properly formatted HTML option list of search conditions. Each query item for which isQueryable() returns true represents a valid search condition. The return value is a string consisting of OPTION tags, suitable to be included between a SELECT tag and its closing tag. If this method is called before connect(), the result is undefined.

Returns: a list of OPTION tags specifying valid query criteria

See Also: isQueryable, connect

showQueryOperators

```
public abstract String showQueryOperators()
throws FrameworkException
```

Returns a properly formatted HTML option list of search operators. The return value is a string consisting of OPTION tags, suitable to be included between a SELECT tag and its closing tag. For queries based on AK regions, the valid operators are AIS, BNOT, CCONTAIN, DSTART, EEND, FGREATER, and GLESS. If this method is called before connect(), the result is undefined.

Returns: a list of OPTION tags specifying valid query operators

Throws: FrameworkException if there is an error while retrieving valid operators (e.g. from the database)

See Also: connect

toURL

```
public abstract String toURL()
```

Externalizes the current state of the query as a URL parameter string of the form "param1=val1¶m2=val2&...¶mX=valX". When creating hyperlinks from one query display page to another, programmers must ensure that the string returned by toURL() is appended to the URL parameters of the link.

5.9 Class QueryCondition

```
java.lang.Object > oracle.apps.ibe.postsales.QueryCondition
```

public abstract class **QueryCondition**

extends Object

QueryCondition encapsulates a single condition in the WHERE clause of a @see Query. The WHERE clause of a Query is a conjunction of zero or more QueryConditions. Note that QueryCondition does not support disjunctions of conditions or joins between database objects at all.

See Also: Query, Region, AkRegionItem

5.9.1 Variables for Class QueryCondition

RCS_ID

```
public static final String RCS_ID
```

RCS_ID_RECORDED

```
public static final boolean RCS_ID_RECORDED
```

5.9.2 Constructors for Class QueryCondition

QueryCondition

```
public QueryCondition()
```

5.9.3 Methods for Class QueryCondition

The following table is an index of Class QueryCondition methods:

Table 5–9 Method Index for Class QueryCondition

Method	Description
getItemIndex	
getOperatorCode	
getValue	

getItemIndex

```
public abstract int getItemIndex()
```

getOperatorCode

```
public abstract String getOperatorCode()
```

getValue

```
public abstract String getValue()
```

5.10 Class QueryUtil

```
java.lang.Object > oracle.apps.ibe.postsales.QueryUtil
```

public final class **QueryUtil**

extends Object

5.10.1 Variables for Class QueryUtil

RCS_ID

```
public static final String RCS_ID
```

RCS_ID_RECORDED

```
public static final boolean RCS_ID_RECORDED
```

5.10.2 Constructors for Class QueryUtil

QueryUtil

```
public QueryUtil()
```

5.10.3 Methods for Class QueryUtil

The following table is an index of Class QueryUtil methods:

Table 5–10 Method Index for Class QueryUtil

Method	Description
boolVal	
doubleVal	
intVal	
isNonNullString	
isNullString	
nonNullString	
stringVal	
validDate	
validNumber	

boolVal

```
public static final boolean boolVal(String paramVal, boolean defaultVal)
```

doubleVal

```
public static final double doubleVal(String paramVal, double defaultVal)
throws FrameworkException
```

intVal

```
public static final int intVal(String paramVal, int defaultVal)
throws FrameworkException
```

isNonNullString

```
public static final boolean isNonNullString(String s)
```

isNullString

```
public static final boolean isNullString(String s)
```

nonNullString

```
public static final String nonNullString(String s)
```

stringVal

```
public static final String stringVal(String paramVal, String defaultVal)
throws FrameworkException
```

validDate

```
public static final boolean validDate(String dateVal, String dateFormat)
```

validNumber

```
public static final boolean validNumber(String numVal)
```

5.11 Class QueryValidatorException

```
java.lang.Object > java.lang.Throwable > java.lang.Exception >
oracle.apps.jtf.base.resources.FrameworkException >
oracle.apps.ibe.postsales.QueryValidatorException
```

public class **QueryValidatorException**

extends FrameworkException

5.11.1 Variables for Class QueryValidatorException

RCS_ID

```
public static final String RCS_ID
```

RCS_ID_RECORDED

```
public static final boolean RCS_ID_RECORDED
```

thisClass

```
public static final String thisClass
```

5.11.2 Constructors for Class QueryValidatorException

QueryValidatorException

```
public QueryValidatorException()
```

QueryValidatorException

```
public QueryValidatorException(String s)
```

QueryValidatorException

```
public QueryValidatorException(String errorKey, Object params[])
```

Construct an exception with the errorKey.

Parameters: params - an array of tokens for errorKey

A
Standards for Customizing JavaServer Pages

This appendix lists the standards you should follow when customizing JavaServer Pages™ (JSP™) for the Oracle iStore 11*i* Web storefront. Topics include

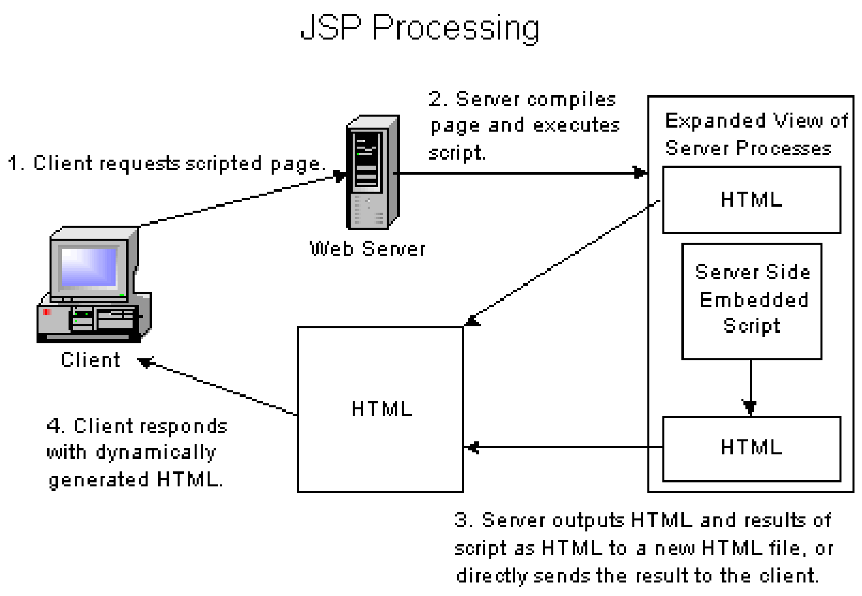
A.1 Understanding JSPs

A JavaServer Page (JSP) is a dynamic HTML Web page that embeds Java language methods in the HTML content to generate dynamic content on the Web page. A JSP file includes HTML, Java, JavaScript, and forms.

A.1.1 JSP Processing

The following diagram illustrates how JSPs are processed.

Figure A–1 JSP Processing

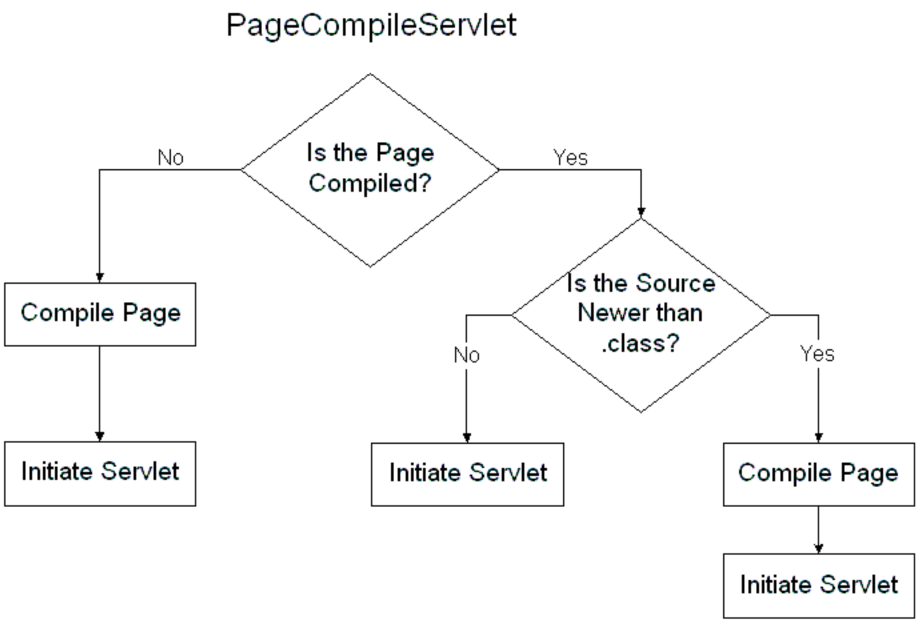

JSP Processing
1. Client requests scripted page.
2. Server compiles page and executes script.
Web Server
Expanded View of Server Processes
HTML
Server Side Embedded Script
HTML
Client
HTML
4. Client responds with dynamically generated HTML.
3. Server outputs HTML and results of script as HTML to a new HTML file, or directly sends the result to the client.

The client browser requests a JSP from the Web server. The server compiles the page and executes the code. The server then creates an HTML page from the JSP's HTML and Java code and sends this page to the client. The client displays the dynamically generated HTML page.

A.1.2 PageCompileServlet

The following diagram illustrates the server's decision process for PageCompileServlet.

Figure A–2 Decision Process for PageCompileServlet



The server checks to see if the page is compiled. If the page is not compiled, then the server compiles the page and initiates the servlet.

If the page is compiled, the server checks to see if the source is newer than the .class file. If the .class file is newer than the source, the server initiates the servlet. If the source is newer than the .class file, the server recompiles the page and then initiates the servlet.

A.1.3 Basic Components of JSPs

JSP files consist of the following basic components.

Directives

- Page Directive

```
<%@ page import="hello.NameHandler" %>
<%@ page info="a hello world example" %>
```

- Include Directive

```
<%@ include file="banner.html" %>
```

Declarations

```
<%!...%>
    <%! int a, b; double c; %>
```

Expressions

```
<%=...%>
    <%= a + b + c %>
```

Scriptlets

```
<%...%>
    <% String name=null;
    if ( request.getParameter("name") == ) { %>
```

Example A–1 foo.jsp

```
<%@ include file="jtfincl.jsp" %>
<%@ include file="ibeCZzpHeader.jsp" %>
<%@pageContext.setAttribute ("_pageTitle", "Test", pageContext.REQUEST_SCOPE);
%>
<%@ include file="ibeCZzpHeader.jsp" %>
Hello World
<%@ include file="ibeCZzdBottom.jsp" %>
```

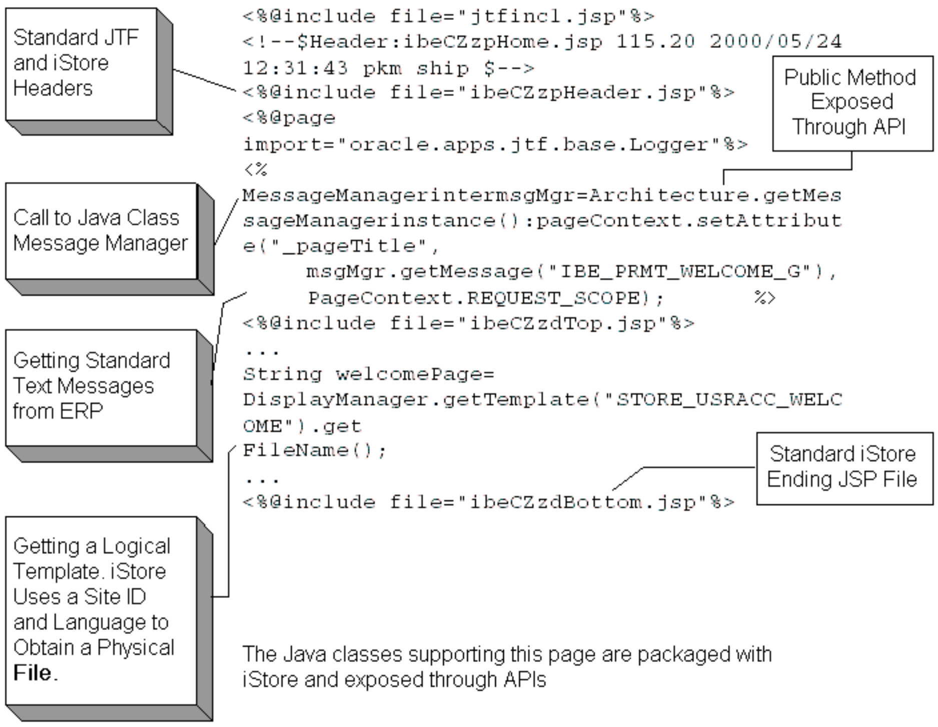
foo.jsp is accessed from a browser as:

```
http://some.domain.com/html/foo.jsp?minisite=10120
```

The minisite ID is a different number for different installations. After the first session, the minisite parameter is passed along in the cookie.

The following figure is a JSP layout example.

Figure A–3 JSP Layout Example



```
<%@include file="jtfincl.jsp"%>
<!--$Header:ibeCZzpHome.jsp 115.20 2000/05/24
12:31:43 pkm ship $-->
<%@include file="ibeCZzpHeader.jsp"%>
<%@page
import="oracle.apps.jtf.base.Logger"%>
<%
MessageManagerintermsgMgr=Architecture.getMes
sageManagerinstance():pageContext.setAttribut
e("_pageTitle",
     msgMgr.getMessage("IBE_PRMT_WELCOME_G"),
     PageContext.REQUEST_SCOPE);       %>
<%@include file="ibeCZzdTop.jsp"%>
...
String welcomePage=
DisplayManager.getTemplate("STORE_USRACC_WELC
OME").get
FileName();
...
<%@include file="ibeCZzdBottom.jsp"%>
```

Standard JTF and iStore Headers
Call to Java Class Message Manager
Getting Standard Text Messages from ERP
Getting a Logical Template. iStore Uses a Site ID and Language to Obtain a Physical **File**.
Public Method Exposed Through API
Standard iStore Ending JSP File
The Java classes supporting this page are packaged with iStore and exposed through APIs

A.1.4 Standard Includes

The Oracle iStore 11*i* standard includes are listed in the following table.

Table A–1 Standard Includes

JSP	Description
jtfincl.jsp	Oracle CRM Technology Foundation standard header
ibeCZzpHeader.jsp	Oracle iStore 11*i* standard header
ibeCZzdTop.jsp	Prints the top of the page to be generated: <head>, <title>, and so on
ibeCZzdMenu.jsp	Prints the top links, tabs, subtabs and the search bar
ibeCZzdBottom.jsp	Standard footer

A.1.5 The Request Object

After a user enters data, the data is stored in the request object, which usually implements javax.servlet.http.HttpServletRequest. You can access the request object from within a scriptlet. The following table lists request objects.

Table A–2 Request Objects

Method	Defined In	Job Performed
getRequest	javax.servlet.jsp.PageContext	Returns the current request object
getParameterNames	javax.servlet.ServletRequest	Returns the names of the parameters that the request currently contains
getParameterValues	javax.servlet.ServletRequest	Returns the values of the parameters that the request currently contains
getParameter	javax.servlet.ServletRequest	Returns the value of the parameter if you provide the name

A.2 JSP Standards (Customer Facing)

Oracle coding standards for Oracle iStore 11*i* Web store JSPs are listed below.

Note: Never modify an original JSP. Make a copy of the original JSP and modify only the copy. If a bug occurs, compare the JSP copy to the JSP original.

Standard Includes for Requested and Forwarded JSPs

These are standard includes for requested and forwarded JSPs, not included JSPs.

Top of JSP

```
<% int pc = PageContext.REQUEST_SCOPE; %>
<% pageContext.setAttribute("_guestNotAllowed", "true", pc);    //WA %>
<% pageContext.setAttribute("_B2CNotAllowed",   "true", pc);    //WA %>
<% pageContext.setAttribute("_permission", "<permission>", pc); //WA %>
<% pageContext.setAttribute("_sensitivePage",   "true", pc);    //WA %>
<%@include file="ibeCZzpHeader.jsp" %>
```

WA: Wherever Applicable

Title or Menu

(not required in "processing only" JSPs)

```
<% MessageManagerInter mm = Architecture.getMessageManagerInstance();
   pageContext.setAttribute("_pageTitle", mm.getMessage("prompt"), pc); %>
<%@include file="ibeCZzdTop.jsp" %>
<!-- say this jsp is for address book. highlight proper tabs/menus -->
<% pageContext.setAttribute("selectedTab",  "account",  pc);
   pageContext.setAttribute("selectedMenu", "userInfo", pc);
   pageContext.setAttribute("selectedSide", "addrbook", pc); %>
<%@include file="ibeCZzdMenu.jsp" %>
<!-- display rest of the page body below this -->
```

Bottom of JSP

(required wherever execution ends)

```
<%@include file="ibeCZzdBottom.jsp" %>
```

Hyperlinks, JSP Include and Forward, and Forms

In the examples below, a template name (for example, STORE_USR_ACC_LOGOUT) is mapped to a raw JSP file name (for example, ibeCAcpLogout.jsp) by the template manager, which is implemented in the DisplayManager class.

Top of JSP Include Page

```
<!-- $Header: standards.html 115.3 2000/06/22 19:04:06 jnath noship $ -->
```

Bottom of JSP Include Page

```
<!-- ibexyyyy.jsp end -->
```

Hyperlink

To display a hyperlink, call the appropriate getURL() method to get the URL. This ensures that the cookie is appended to the hyperlink, in case the browser is not accepting cookies.

```
DisplayManager.getURL(<template name>)
DisplayManager.getURL(<template name>, <query string>)
RequestCtx.getURL(<jsp filename>)
RequestCtx.getURL(<jsp filename>, <query string>)
```

For example,

```
<% String url = DisplayManager.getURL("STORE_USR_ACC_LOGOUT"); %>
<a href=<%=url%>Click here to logout</a>
```

File Name (for JSP Include, JSP Forward, and Form Actions)

For JSP include, JSP forward and form actions, the raw JSP file name is required instead of the URL. Call `DisplayManager.getTemplate` to get the JSP file name from the template name.

For example,

```
<% String jspfilename = DisplayManager.getTemplate(
                        "STORE_SOME_TEMPLATE").getFileName() %>
<jsp:include page="<%=jspfilename%>" flush="true"/> OR
<jsp:forward page="<%=jspfilename%>" />
```

Forms

In a form, apart from getting the JSP file name from the template name to be used as the action, also call `RequestCtx.getSessionInfoAsHiddenParam()` within the form. This ensures that the cookie is passed as a hidden field, in case the browser is not accepting cookies.

For example,

```
<form method=post action=<%=jspfilename%>>
  ...
  <%= RequestCtx.getSessionInfoAsHiddenParam() %>
</form>
```

A.2.1 Transactions and Database Connections

This section displays the JSP and Java API code for transactional APIs and read-only APIs.

Transactional APIs: JSP Code

```
Object txnLock = new Object();
try {
  TransactionScope.begin(txnLock);
  Call transactional and other APIs
}
catch (CustomerException e) {  // exceptions that can be handled
  TransactionScope.setRollbackOnly();
  errorFlag = true;
}
catch (Exception e1) {         // other unexpected exceptions
  TransactionScope.setRollbackOnly();
  throw e1;
}
finally {
  TransactionScope.end(txnLock);
}
if (errorFlag == true) {
  handle error: display message or jsp:forward or sendRedirect
  if sendRedirect, call RequestCtx.end() first
}
```

Transactional APIs: Java API Code

```
public static void transactionalApi()
  throws FrameworkException, CustomerException, SQLException
{
  OracleConnection conn = (OracleConnection) TransactionScope.getConnection();
  Statement        stmt = ... ;
  ...
  try {
    stmt.execute();
    ...
  }
  finally {
    if (stmt != null) stmt.close();
  }
}
```

Read-Only APIs: JSP Code

```
try {
  readonlyApi();
  readonlyApi();
}
catch (CustomerException e) {  // catch exceptions that can be handled
  // handle exception
}
```

Read-Only APIs: Java API Code

```
public static void readonlyApi()
  throws FrameworkException, SQLException
{
  OracleConnection conn = (OracleConnection) TransactionScope.getConnection();
  Statement        stmt = ... ;
  ...
  try {
    stmt.execute();
    ...
  }
  finally {
    if (stmt != null) stmt.close();
    TransactionScope.releaseConnection(conn);
  }
}
```

A.2.2 Exception Handling

You can set up exception handling for user errors and fatal errors.

User Errors

Raise a subclass of FrameworkException in the API and handle it in the calling JSP.

Fatal Errors

Raise FrameworkException or let the original exception (for example, SQLException) propagate to ibeCZzdError.jsp.

A.2.3 API Standards

Following are the API standards for package naming, load methods, accessing member variables, get methods, and standard parameters.

Package Naming

Customer Side

```
oracle.apps.ibe.<sub-component>
```

or

```
oracle.apps.ibe.<sub-component>.<sub-sub-component>
```

Merchant Side

```
oracle.apps.ibe.setup.<sub-component>
```

Load Method(s)

Each class will have at least one static load method. Given a recordId, it gets the record from the database or cache. The API returns the record object. The method either caches the record object (for certain classes) or performs deep load or shallow load (for certain classes).

For example,

```
Product prd = Product.load(productid, "DEEP");
Account acc = Account.load(accountid);
```

Accessing Member Variables

Once a record object is loaded, the member variables can be accessed directly, for example, `acc.accountId`.

They can also be accessed using get methods, for example, `prd.getProductId()`.

Get Methods

Get methods are not mandatory, except for translatable attributes. It will return the value in the appropriate language, for example, `prd.getDescription()`.

Standard Parameters

Standard parameters, such as user ID and language code, will be available statically from RequestCtx, for example, `RequestCtx.getUserId()`.

A.3 Passing Values Across Templates

Use the following guidelines for passing values across templates.

pageContext.setAttribute

```
pageContext.setAttribute("_pageTitle",
            "iStore Framework",
            PageContext.REQUEST_SCOPE);
```

pageContext.getAttribute

```
pageContext.getAttribute("_pageTitle",
            PageContext.REQUEST_SCOPE);
```

Example Parameters

```
<%@ include file="jtfincl.jsp" %>

<% int pc = PageContext.REQUEST_SCOPE; %>
<% pageContext.setAttribute("_guestNotAllowed", "true", pc);// #1
<% pageContext.setAttribute("_B2CNotAllowed","true", pc);// #2
<% pageContext.setAttribute("_permission", "<permission>", pc;// #3
<% pageContext.setAttribute("_sensitivePage", "true", pc);// #4
<%@ include file="ibeCZzpHeader.jsp" %>

<% pageContext.setAttribute("_pageTitle", "iStore Framework", pc); %>
<%@ include file="ibeCZzdTop.jsp" %>

pageContext.setAttribute("selectedTab", "0", pc);// #5
pageContext.setAttribute("selectedMenu", "0", pc);// #6
pageContext.setAttribute("selectedSide", "0", pc);// #7
<%@ include file="ibeCZzdMenu.jsp"%>

Hello World
<%@ include file="ibeCZzdBottom.jsp" %>
```

See the following table for notes about this code example.

Table A–3 Example Notes

Note	Description
# 1	▪ Ensures that only logged in users can access the page and see the "Hello World" message. ▪ Anonymous users are asked to sign in or register.
# 2	Ensures that only registered business customers can access the page.
# 3	▪ To check permissions, for example, if foo.jsp is used to create an address, then <permission> is IBE_CREATE_ADDRESS. ▪ Oracle iStore 11*i* ships with predefined permissions that you can use to restrict the access of B2B users to a page. For B2C users, Oracle iStore 11*i* does not check permissions, and instead checks the _B2CNotAllowed flag.
# 4	▪ Controls access to sensitive pages, for example, users' personal information such as email address, credit card number, and so on. ▪ The product catalog and view shopping cart pages are non-sensitive pages. When users go from a non-sensitive page to a sensitive page, they are re-authenticated. ▪ Sensitive pages should be SSL enabled. In general, all _ guestNotAllowed pages are sensitive pages.
# 5, # 6, # 7	▪ Specifies which tab, subtab, and side menu to highlight, starting from 0. For example, see ibeCZzdMenu.jsp.

A.4 Passing Parameters or Cookies

To pass parameters or cookies, use the following guidelines.

See Example Parameters and replace “Hello World” with the following line:

```
<a href=<%=DisplayManager.getURL("STORE_HOME")%>>Go to iStore Home</a>
```

Display Manager then calls RequestCtx.getURL(<jspFileName>) to append the cookie string to the JSP file name if necessary.

To pass more parameters, use

```
DisplayManager.getURL (<accessName>, <query string>) or
RequestCtx.getURL (<jspFileName>, <query string>)
```

where the query string is of the form: "`foo1=bar1&foo2=bar2&foo3=bar3`".

In a form, the cookie string is passed as a hidden field, for example,

```
RequestCtx.getSessionInfoAsHiddenParam()
```

To access a physical file without a cookie, use

```
DisplayManager.getTemplate (<access name>) .getFileName()
```

A.5 Using Forms with JSPs

To use forms with JavaServer Pages, use the following procedure.

Prerequisites

None.

Steps

1. Create an HTML form in a JSP template to process user input.
2. Perform basic checks on the input that the user enters, for example, using JavaScript.
3. Pass the data to a bean that implements the business logic.

   The bean processes the input (it can maintain a persistent state), and returns any results back to the user as HTML.

A.6 Using JavaBeans with JSPs

To use JavaBeans with JavaServer Pages, use the following guidelines.

<jsp:useBean>

Instantiate or locate the JavaBean instance.

```
<jsp:useBean id="mybean" scope="session"
class="hello.nameHandler" />
```

<jsp:setProperty>

Set property values in the JavaBean.

```
<jsp:setProperty name="mybean" property="*" />
```

<jsp:getProperty>

Get property values from the JavaBean.

```
<jsp:getProperty name="mybean" property="username" />
```

A.7 Adding Error Pages

To add error pages, use the following procedure.

Steps

1. Write the JavaBean (or servlet or other component) so that it throws certain exceptions under certain conditions.
2. In the JSP file, use a page directive with errorPage set to the name of a JSP file that will display a message to the user when the exception occurs.
3. Write an error page file using a page directive with isErrorPage="true." For example, see ibeCZzdError.jsp.
4. In the error page file, use the exception object to get information about the exception.
5. Give informative messages in the error file or the JSP file on the error.

A.8 Handling Runtime Exceptions

To handle runtime exceptions, use the following guidelines.

User Errors

To handle user errors, raise a subclass of FrameworkException in the API and handle it in the calling JSP.

Fatal Errors

To handle fatal errors, raise FrameworkException or let the original exception (for example, SQLException) propagate to ibeCZzdError.jsp.

B Setting Up Oracle JDeveloper

This appendix explains how to set up Oracle JDeveloper to write, debug, deploy, and test custom Java code and JSPs for your Oracle iStore 11*i* implementation. Topics include:

B.1 Overview of Oracle JDeveloper

Oracle JDeveloper is a development environment using Java 2 Enterprise Edition™ (J2EE™) and Extensible Markup Language (XML) with full support for developing, debugging, and deploying business applications and Web services.

Oracle JDeveloper offers highly productive development tools, including a Java debugger and the innovative profiler and CodeCoach tools for code performance analysis and improvement. Oracle JDeveloper includes wizards, editors, and visual design tools for developing applications and Web services that use applets, applications, JavaBeans, JavaServer Pages, servlets, and Enterprise JavaBeans.

Taking application development to a higher level of productivity, Oracle JDeveloper offers integrated UML modeling, software configuration management, and a J2EE design patterns framework called Business Components for Java (BC4J).

B.2 Setting Up Oracle JDeveloper

You can use Oracle JDeveloper as a development environment to create custom Java code and JSPs for your Oracle iStore 11*i* implementation.

Use the following procedure to set up Oracle JDeveloper to run the Oracle iStore 11*i* middle tier in a Microsoft Windows environment.

Prerequisites

- Oracle JDeveloper 3.2.3 is installed on a Windows NT machine. See the Oracle JDeveloper documentation for instructions.
- Oracle iStore 11*i* is correctly set up on a UNIX server. See *Oracle iStore Implementation Guide* for instructions.

Steps

1. Build the Directory Structure.
2. Set Up the Workspace and Project.
3. Set Up the Database Connection.
4. Run Oracle iStore 11i from Oracle JDeveloper.

B.2.1 Build the Directory Structure

After installing Oracle JDeveloper, you must create an appropriate directory structure for setting up a workspace, a project, and source code. You can set up the directory structure in several ways.

The following procedure describes the recommended directory structure setup.

Steps

1. Create the directory `C:\Oracle\iStore` on the Windows NT machine to store the source code.
2. Create the directories listed in Table B–1. Add the source code files to the appropriate directories.

   Note: All .java files can reside in the same directory even if they belong to different packages.

3. To run the Oracle iStore 11*i* Customer UI, copy the Customer UI files listed in Table B–2 from the UNIX server to the Windows NT machine.
4. To run the Oracle iStore 11*i* Merchant UI, copy the Merchant UI files listed in Table B–3 from the UNIX server to the Windows NT machine, in addition to the files listed in Table B–2.
5. To use Oracle Marketing Online's eMerchandising postings, copy ibapstng.jsp and iba*.xml to the HTML directory (`C:\Oracle\iStore\html`).

Guidelines

The following table lists the source code directories to create and the files you should add to them.

Table B–1 Source Code Directories

Directory Type	Directory	File Types Stored
HTML directory	`C:\Oracle\iStore\html`	css, html, js, jsp, xml
Java directory	`C:\Oracle\iStore\java`	java, zip, jar
PL/SQL directory	`C:\Oracle\iStore\sql`	sql, pls
Output directory	`C:\Oracle\iStore\class`	class
Configuration directory	`C:\Oracle\iStore\config`	dbc

Table B–1 Source Code Directories (Cont.)

Directory Type	Directory	File Types Stored
Media directory	`C:\Oracle\iStore\OA_MEDIA`	gif, jpg
Custom HTML directory	`C:\Oracle\iStore\custom_html`	css, html, js, jsp, xml

The following table lists the Customer UI files you should copy from the UNIX server to the Windows NT machine.

Table B–2 Oracle iStore 11i Customer UI Files

Files to Copy	Source Directory (UNIX)	Destination Directory (Windows NT)
ibeC*, jtf*	`$APPL_TOP/html`	`C:\Oracle\iStore\html`
ibe*, jtf*	`$APPL_TOP/media`	`C:\Oracle\iStore\OA_MEDIA`
apps.zip	`$APPL_TOP/java`	`C:\Oracle\iStore\java`

The following table lists the Merchant UI files you should copy from the UNIX server to the Windows NT machine in addition to the files listed in Table B–2, "Oracle iStore 11i Customer UI Files".

Table B–3 Oracle iStore 11i Merchant UI Files

Files to Copy	Source Directory (UNIX)	Destination Directory (Windows NT)
ibe*, jtf*	`$APPL_TOP/html`	`C:\Oracle\iStore\html`

B.2.2 Set Up the Workspace and Project

When you launch Oracle JDeveloper, it usually opens a workspace and project that you can use. If not, then use the following procedure to set up the workspace and project.

Prerequisites

You have performed the procedure described in Section B.2.1, "Build the Directory Structure".

Steps

1. Launch Oracle JDeveloper.
2. Create a new workspace as follows:

   a. Choose **File > New Workspace.**
   b. Select the default workspace name.
   c. Choose **File > Rename.**
   d. Give the workspace an appropriate name. This procedure assumes that you have named the workspace Welcome.jws.

3. Create a new project as follows:
   a. Choose **File > New Empty Project.**

      The default project name appears under the workspace.
   b. Select the default project name.
   c. Choose **File > Rename.**
   d. Give the project an appropriate name. This procedure assumes that you have named the workspace Welcome.jpr.
4. Right-click on Welcome.jpr and choose **Properties** from the menu.
5. In the Path tab, enter the following values:
   - Target JDK version = **JDK 1.1.8**
   - Source root directories = `C:\Orant\iStore`
   - Output root directory = `C:\Oracle\iStore\class`
   - Run/Debug working directory = `C:\Oracle\iStore\class`
   - HTML root directory = `C:\Oracle\iStore`
6. Run the test described in Section B.3.1, "Create, Compile, and Run a Test JSP Program" to check that your setup is correct up to this point.
7. In the Libraries tab, include **JDeveloper Runtime, Oracle 8.1.7 JDBC, Connection Manager,** and **JSP Runtime** in the list of Java libraries.
8. In the Libraries tab, include a new library called **iStore Library** in the list of Java libraries as follows:
   a. Click **Add.**
   b. In the popup window, click **New.**
   c. Enter the library's name as `iStore Library`.
   d. Enter the library's class path as `C:\Oracle\iStore\java\apps.zip`.

   e. Click **OK** to return to the Libraries tab.
   f. Move **iStore Library** to the top of the Java library list.

9. In the Run/Debug tab, set the following values:

   - Debug Files as = **Normal Java class**
   - Parameters = `intra=t`
   - Java VM Executable = **bcwdbkjv.exe** (This should be the default.)
   - Java VM Parameters =

     ```
     -mx50m -DJTFDBCFILE=C:\Oracle\iStore\config\iStore.dbc
     -Dframework.Logging.system.filename=C:\Temp\fwsys.log
     -Dservice.Logging.common.filename=C:\Temp\istore.log
     -Djtt_cookie_path=/ -Djtt_cookie_domain=<cookie domain>
     ```

     <cookie domain> is the domain for which Oracle iStore 11*i* cookies will be set. This could be the IP address or domain of the Windows NT machine. Set <cookie domain> to the domain that appears in the URL in the test described in Section B.3.1, "Create, Compile, and Run a Test JSP Program".

   - Compile project before running or debugging = **No <unchecked>**

10. Save your changes.

B.2.3 Set Up the Database Connection

The database configuration (.dbc) file for an environment contains the database connection information that enables the middle-tier application to access the database. Since the database instance is the same for both UNIX and desktop middle tiers, you can use the .dbc file from the UNIX middle tier in a Windows NT environment.

The following is an example of a .dbc file:

```
APPL_SERVER_ID=2E16F3BCE7BF673DE03400400B40874620439479001964928795210957097215
TWO_TASK=istore1151
GUEST_USER_PWD=GUEST/GUEST
FNDNAM=apps
GWYUID=applsyspub/pub
APPS_JDBC_DRIVER_TYPE=THIN
DB_HOST=xyz.dot.com
DB_PORT=1521
APPS=apps/apps
```

Use the following procedure to set up the database connection.

Prerequisites

You have performed the procedures described in Section B.2.1, "Build the Directory Structure" and Section B.2.2, "Set Up the Workspace and Project".

Steps

1. Retrieve the .dbc file from the UNIX middle tier where you have installed Oracle iStore 11*i*.
2. Rename the .dbc file to iStore.dbc.
3. Copy the iStore.dbc file to the Configuration directory (`C:\Oracle\iStore\config`).
4. Run the test described in Section B.3.2, "Test Database Access" to check that your database connection is set up correctly.

B.2.4 Run Oracle iStore 11*i* from Oracle JDeveloper

You can now run Oracle iStore 11*i* from Oracle JDeveloper in your Windows NT environment.

It is strongly recommended that you compile JSPs before accessing them from your browser. If you do not pre-compile the JSPs, the response will be very slow.

The compilation method depends on the JSP type. There are three types of JSP files:

- **Top-level JSPs—**You can compile and access top-level JSPs from the browser by entering the URL. Examples of top-level JSPs include ibeCZzpEntry.jsp, ibeCZzdMinisites.jsp, ibeCZzpHome.jsp, and ibeCAcdLogin.jsp.
- **Compile-time included JSPs—**These are included from other JSPs with the tag `<%@ include file="include_1.jsp" %>` and are equivalent to a macro. They cannot be compiled independently. Examples of compile-time JSPs include ibeCZzpHeader.jsp, ibeCZzpTop.jsp, ibeCZzpMenu.jsp, and ibeCZzdBottom.jsp.
- **Run-time included JSPs—**These are included from other JSPs with the tag `<jsp:include page="<%=include_2.jsp%>" flush="true" />` and are equivalent to a function call. They can be compiled independently. Examples of run-time JSPs include ibeCAcdWelcome.jsp and ibeCCtdSctBrwsBin.jsp.

You do not need to add and compile all top-level and run-time included JSPs before running Oracle iStore 11*i*. You can compile only those JSPs that are required to bring up a certain page on the browser. If a JSP that is needed to bring up a page is not pre-compiled, it will be compiled automatically at run-time, but the response will be very slow.

Use the following procedure to run Oracle iStore 11*i* from Oracle JDeveloper.

Prerequisites

You have performed the procedures described in Section B.2.1, "Build the Directory Structure", Section B.2.2, "Set Up the Workspace and Project", and Section B.2.3, "Set Up the Database Connection".

Steps

1. Launch Oracle JDeveloper.
2. Add ibeCAcdLogin.jsp to your project by highlighting Welcome.jpr and choosing **File > Open.**
3. Compile and run ibeCAcdLogin.jsp.

   The browser should launch and display the login page.

   Note: The entry pages to the Oracle iStore 11*i* Customer UI are ibeCZzpEntry.jsp, ibeCZzdMinisites.jsp and ibeCZzpHome.jsp. However, these pages may redirect to other pages, depending on application configurations. So, for the simplest approach, bring up the login page first.

4. Add top-level and run-time included JSP files to the Welcome.jpr project by highlighting Welcome.jpr and choosing **File > Open.**
5. Compile all of the files in the project as follows:
   a. Highlight Welcome.jpr and choose **Project > Make.**

      Oracle JDeveloper compiles the files. If you have added compile-time included JSPs to the project, then compilation errors will result, since compile-time included JSPs cannot compile independently.

   b. Optional: If compilation errors occur, compile each top-level and run-time included JSP individually by right-clicking on the JSP file name and choosing **Make** from the menu.

6. Run the JSPs that you want to view.

B.3 Testing the Oracle JDeveloper Setup

Use the following procedures to test your Oracle JDeveloper setup:

- Create, Compile, and Run a Test JSP Program
- Test Database Access

B.3.1 Create, Compile, and Run a Test JSP Program

Use the following procedure to create, compile, and run a test JSP program.

Steps

1. Launch Oracle JDeveloper.
2. Create a new JSP by choosing **File > New > Web Objects > JSP.**
3. Keep the default text or create one line: "`Hello World!`"
4. Save the JSP as Test.jsp in the HTML directory (`C:\Oracle\iStore\html`) by choosing **File > Save As.**
5. In Oracle JDeveloper, add Test.jsp to your project by highlighting Welcome.jpr and choosing **File > Open.**
6. Compile Test.jsp by right-clicking on Test.jsp and choosing **Make** from the menu.

   Test.jsp should compile successfully.
7. Run Test.jsp by right-clicking on Test.jsp and choosing **Run** from the menu.

   A browser should launch and display, "Hello World!".

B.3.2 Test Database Access

Use the following procedure to check that the Windows NT middle tier can access the database.

Prerequisites

You have created Test.jsp as described in Section B.3.1, "Create, Compile, and Run a Test JSP Program".

Steps

1. Modify Test.jsp with the following code:

   ```
   <%@page import="oracle.apps.jtf.base.session.FWAppsContext" %>
   <%@page import="java.sql.*" %>

   <%
   FWAppsContext ctx  =
      new FWAppsContext("C:\\Oracle\\iStore\\config\\iStore.dbc");
   Connection    conn = ctx.getConnection();
   Statement     stmt = conn.createStatement();
   ResultSet     rset = stmt.executeQuery("select global_name from global_
   name");

   rset.next();
   out.println(rset.getString(1));
   rset.close();
   stmt.close();
   conn.close();
   %>
   ```

2. Compile Test.jsp by right-clicking on Test.jsp and choosing **Make** from the menu.

   Test.jsp should compile successfully.

3. Run Test.jsp by right-clicking on Test.jsp and choosing **Run** from the menu.

   A browser should launch and display the database global name.

B.4 Customization Guidelines

Follow these guidelines as you customize your Oracle iStore 11*i* implementation using Oracle JDeveloper:

- To customize the Customer UI, copy files from the HTML directory to the Custom HTML directory so that you do not modify the seeded files in the HTML directory. The JSPs that you would need to customize are named ibeC*.jsp.
- Put new custom JSP files in the Custom HTML directory.
- Put any custom Java and PL/SQL files in their respective directories.
- Create different folders in your Oracle JDeveloper project to organize source files, by right-clicking on Welcome.jpr and choosing **Add Folder** from the menu.
- If you are customizing compile-time included JSPs, keep them in one folder, such as "Run Time," and top-level and run-time included JSPs in another folder, such as "Top Level." You can compile all top-level and run-time included JSPs together by right-clicking on Top Level and choosing **Make** from the menu.