



InformixEOAdaptor Framework

Java API Reference



Apple Computer, Inc.
© 1999 Apple Computer, Inc.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Computer, Inc., except to make a backup copy of any documentation provided on CD-ROM.

The Apple logo is a trademark of Apple Computer, Inc. Use of the “keyboard” Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this book. Apple retains all intellectual property rights associated with the technology described in this book. This book is intended to assist application developers to develop applications only for Apple-labeled or Apple-licensed computers.

Every effort has been made to ensure that the information in this manual is accurate. Apple is not responsible for typographical errors.

Apple Computer, Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Apple, the Apple logo, Macintosh, and WebObjects are trademarks of Apple Computer, Inc., registered in the United States and other countries. Enterprise Objects is a trademark of Apple Computer, Inc.

NeXT, the NeXT logo, OPENSTEP, Enterprise Objects Framework, Objective-C, and WEBSOCKET are trademarks of NeXT Software, Inc.

Adobe, Acrobat, and PostScript are trademarks of Adobe Systems Incorporated or its subsidiaries and may be registered in certain jurisdictions.

Helvetica and Palatino are registered trademarks of Linotype-Hell AG and/or its subsidiaries.

ITC Zapf Dingbats is a registered trademark of International Typeface Corporation.

ORACLE is a registered trademark of Oracle Corporation, Inc.

SYBASE is a registered trademark of Sybase, Inc.

UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company Limited.

Windows NT is a trademark of Microsoft Corporation.

All other trademarks mentioned belong to their respective owners.

Simultaneously published in the United States and Canada.

Even though Apple has reviewed this manual, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS MANUAL, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS MANUAL IS SOLD “AS IS,” AND YOU, THE PURCHASER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS MANUAL, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

InformixEOAdaptor

Package: `com.apple.yellow.informixeadaptor`

Introduction

The InformixEOAdaptor framework is a set of classes that allow your programs to connect to an Informix server. These classes provide Informix-specific method implementations for the EOAccess framework's EOAdaptor, EOAdaptorChannel, EOAdaptorContext, and EOSQLExpression abstract classes.

The following table lists the classes in the InformixEOAdaptor Framework and provides a brief description of each class.

Class	Description
InformixAdaptor	Represents a single connection to a Informix database server, and is responsible for keeping login and model information, performing Informix-specific formatting of SQL expressions, and reporting errors.
InformixChannel	Represents an independent communication channel to the database server its InformixAdaptor is connected to.
InformixContext	Represents a single transaction scope on the database server to which its adaptor object is connected.
InformixSQLExpression	Defines how to build SQL statements for InformixChannels.

The Connection Dictionary

The connection dictionary contains items needed to connect to an Informix server, such as the database name (it's common to omit the user name and password from the connection dictionary, and prompt users to enter those values in a login panel). The keys of this dictionary identify the information the server expects, and the values of those keys are the values that the adaptor uses when trying to connect to the server.

The Informix adaptor defines string constants for use as connection dictionary keys:

- `dbNameKey`
- `userNameKey`
- `passwordKey`

See the `InformixAdaptor` class specification for more information on the connection dictionary key constants.

Locking

All adaptors use the database server's native locking facilities to lock rows on the server. In the Informix adaptor locking is determined by the isolation level, which is implemented in `InformixChannel`. Locking occurs when:

- You send the adaptor channel a `selectAttributes` message with `true` specified as the value for the `lock` parameter.
- You explicitly lock an object's row with the `EODatabaseContext`'s `lockObjectWithGlobalID` message.
- You set pessimistic locking at the database level and fetch objects.

Data Type Mapping

Every adaptor provides a mapping between each server data type and the Java type to which a database value will be coerced when it's fetched from the database. The following table lists the mapping used by InformixAdaptor.

Informix Data Type	Java Data Type
VARCHAR	String
NVARCHAR	String
DECIMAL	BigDecimal
MONEY	BigDecimal
BYTE	NSData
TEXT	String
DATE	NSGregorianCalendar
INTEGER	Number
SMALLINT	Number
NCHAR	String
CHAR	Number
SERIAL	Number
FLOAT	Number
SMALLFLOAT	Number
DATETIME YEAR TO SECOND	NSGregorianCalendar
INTERVAL	String

The type mapping methods—`externalTypesWithModel`, `internalTypeForExternalTypeInModel`, and `isValidQualifierTypeInModel`—allow for an adaptor to supplement its set of type mappings with additional mappings for user-defined database types. InformixAdaptor does not make use of the model argument if one is provided.

Prototype Attributes

The InformixEOAdaptor Framework provides the following set of prototype attributes:

Name	External Type	Value Class Name	Other Attributes
binaryID	BYTE	NSData	
city	VARCHAR	String	columnName = CITY width = 50
date	DATETIME YEAR TO SECOND	NSDate	columnName = ""
longText	TEXT	String	
money	INTEGER	BigDecimal	columnName = ""
phoneNumber	VARCHAR	String	columnName = PHONE width = 20
rawImage	BYTE	NSData	columnName = RAW_IMAGE
state	VARCHAR	String	columnName = STATE width = 2
streetAddress	VARCHAR	String	columnName = STREET_ADDRESS width = 100

Name	External Type	Value Class Name	Other Attributes
tiffImage	BYTE	NSImage	adaptorValueConversionMethodName = TIFFRepresentation columnName = PHOTO valueFactoryMethodName = "imageWithData:"
uniqueID	INTEGER	Number	columnName = "" valueType = i
zipCode	VARCHAR	String	columnName = ZIP width = 10

Generating Primary Keys

Each adaptor provides a database-specific implementation of the method `primaryKeyForNewRow` for generating primary keys. The `InformixChannel`'s implementation uses a table named `eo_sequence_table` to keep track of the next available primary key value for a given table. The table contains a row for each table for which the adaptor provides primary key values. The statement used to create the `eo_sequence_table` is:

```
create table eo_sequence_table (
    table_name varchar(32,0),
    counter integer
)
```

`InformixChannel` uses a stored procedure named `eo_pk_for_table` to access and maintain the primary key counter in `eo_sequence_table`. The stored procedure is defined as follows:

```
create procedure
eo_pk_for_table (tname varchar(32))
returning int;
    define cntr int;

    update EO_SEQUENCE_TABLE
    set COUNTER = COUNTER + 1
    where TABLE_NAME = tname;
```

FRAMEWORK InformixEOAdaptor

```
select COUNTER into cntr
from EO_SEQUENCE_TABLE
where TABLE_NAME = tname;

return cntr;
end procedure;
```

The stored procedure increments the counter in the `eo_sequence_table` row for the specified table, selects the counter value, and returns it. `InformixChannel` executes this `eo_pk_for_table` stored procedure from `primaryKeyForNewRow` and returns the stored procedure's return value.

To use `InformixChannel`'s database-specific primary key generation mechanism, be sure that your database accommodates the adaptor's scheme. To modify your database so that it supports the adaptor's mechanism for generating primary keys, use `EOModeler`. For more information on this topic, see *Enterprise Objects Framework Developer's Guide*.

Bind Variables

The `InformixAdaptor` uses bind variables. A bind variable is a placeholder used in an SQL statement that is replaced with an actual value after the database server determines an execution plan. You use the following methods to operate on bind variables:

- `bindValueDictionaryForAttribute`
- `mustUseBindVariableForAttribute`
- `shouldUseBindVariableForAttribute`

InformixAdaptor

Inherits from: EOAdaptor : NSObject

Package: com.apple.yellow.informixeadaptor

Class Description

An InformixAdaptor represents a single connection to an Informix database server, and is responsible for keeping login and model information, performing Informix-specific formatting of SQL expressions, and reporting errors.

The InformixAdaptor doesn't support full outer joins.

Constants

InformixAdaptor defines the following string constants for use as connection dictionary keys.

Constant	Corresponding value in the connection dictionary
dbNameKey	The name of the database.
userNameKey	The name of the user to log in as.
passwordKey	The user's password.

CLASS InformixAdaptor

It defines a string constant for use as a key in an exception's userInfo dictionary (see `raiseInformixError`).

Constant	Corresponding value in an exception's userInfo dictionary
<code>InformixErrorKey</code>	The error code raised in the Informix server.

`InformixAdaptor` also defines a string constant to identify the user defaults domain for the Informix adaptor.

Constant	Description
<code>EOF_INFORMIX_ADAPTOR</code>	The name of the user defaults domain for the Informix adaptor.

Method Types

Mapping external types to internal types

`externalToInternalTypeMap`

Getting information from the connection dictionary

`informixConnectionString`

`informixDefaultForKey`

`informixPassword`

`informixUserName`

`connectionKeys`

Error handling

`raiseInformixError`

Preparing to connect

`prepareEnvironmentForConnect`

`resetEnvironmentAfterConnect`

CLASS InformixAdaptor

Callback methods

`informixContextDidDisconnect`

`informixContextWillConnect`

Getting adaptor-specific classes

`adaptorContextClass`

`adaptorChannelClass`

Static Methods

externalToInternalTypeMap

```
public static NSDictionary externalToInternalTypeMap()
```

Returns the mapping between each predefined external (database) type known by the adaptor to a default internal type. For information on the mapping, see the section in the InformixEOAdaptor Framework introduction titled [“Data Type Mapping”](#) (page 5).

Instance Methods

adaptorChannelClass

```
public Class adaptorChannelClass()
```

Returns the InformixChannel class.

adaptorContextClass

```
public Class adaptorContextClass()
```

Returns the InformixContext class.

CLASS InformixAdaptor

connectionKeys

```
public NSArray connectionKeys()
```

Returns an NSArray containing the keys in the receiver's connection dictionary. You can use this method to prompt the user to supply values for the connection dictionary.

informixConnectionString

```
public String informixConnectionString()
```

Returns the user name, password, and database name as a string suitable to be supplied as an argument to `db_connect()`.

informixContextDidDisconnect

```
public void informixContextDidDisconnect(InformixContext logon)
```

Callback method that is invoked after the associated Informix context disconnects.

informixContextWillConnect

```
public void informixContextWillConnect(InformixContext logon)
```

Callback method that is invoked just before the associated Informix context disconnects.

informixDefaultForKey

```
public String informixDefaultForKey(String key)
```

Returns the user default setting for *key*. To get this information it first checks the user defaults, and then the adaptor's internal defaults dictionary.

informixPassword

```
public String informixPassword()
```

Returns the password in the connection dictionary.

CLASS InformixAdaptor

informixUserName

```
public String informixUserName()
```

Returns the user name in the connection dictionary.

prepareEnvironmentForConnect

```
public void prepareEnvironmentForConnect()
```

Prepares the user environment for connection to an Informix server. Preserves existing environment variables, replacing them with values obtained from the adaptor's connection dictionary. Unset variables are set to values obtained from `informixDefaultForKey`, if any.

See Also: `resetEnvironmentAfterConnect`

raiseInformixError

```
public void raiseInformixError(String sqlString)
```

Examines Informix structures for error flags and raises an exception if one is found. Extracts the error information in the connection structure and uses it to build and raise an exception. The error code is available in the exception's user info dictionary in the entry for the key, `InformixErrorKey`.

resetEnvironmentAfterConnect

```
public void resetEnvironmentAfterConnect()
```

Restores any environment variables overwritten by `prepareEnvironmentForConnect`.

CLASS InformixAdaptor

InformixChannel

Inherits from: EOAdaptorChannel : NSObject

Package: com.apple.yellow.informixeadaptor

Class Description

An InformixChannel represents an independent communication channel to the database server its InformixAdaptor is connected to. All of an InformixChannel's operations take place within the context of transactions controlled or tracked by its InformixContext. An InformixContext can manage multiple InformixChannels, and a channel is associated with only one context.

The features InformixChannel adds to EOAdaptorChannel are as follows:

- Informix-specific error handling (see InformixChannel.Delegate)
- The ability to configure the fetch buffer
- The ability to read a list of table names from the database

Method Types

Finding table names

```
setInformixTableNamesSQL
```

CLASS InformixChannel

```
informixTableNamesSQL
```

Setting the fetch buffer length

```
setFetchBufferLength
```

```
fetchBufferLength
```

Instance Methods

fetchBufferLength

```
public int fetchBufferLength()
```

Returns the size, in bytes, of the fetch buffer. The larger the buffer, the more rows can be returned for each round trip to the server.

informixTableNamesSQL

```
public String informixTableNamesSQL()
```

Returns the SQL statement the receiver uses to find table names. The user default `InformixTableNamesSQL` overrides a statement set with `setInformixTableNamesSQL`.

setFetchBufferLength

```
public void setFetchBufferLength(int length)
```

Sets the size (in bytes) of the fetch buffer to *length*. The larger the buffer, the more rows can be returned for each round trip to the server.

setInformixTableNamesSQL

```
public void setInformixTableNamesSQL(String sql)
```

Set the SQL statement the receiver uses to find table names to *sql*.

InformixContext

Inherits from: EOAdaptorContext : NSObject
Package: com.apple.yellow.informixeadaptor

Class Description

An InformixContext represents a single transaction scope on the database server to which its adaptor object is connected. If the server supports multiple concurrent transaction sessions, the adaptor may have several adaptor contexts. An InformixContext may in turn have several InformixChannels, which handle actual access to the data on the server.

The features the InformixContext class adds to EOAdaptorContext are methods for setting Informix-specific characteristics for the context.

Method Types

Managing a connection to the server

- connect
- connection
- disconnect

CLASS InformixContext

isConnected

Returning information about an InformixContext

fetchesInProgress

hasTransactions

Returns information about the server

isOnLine

Instance Methods

connect

```
public void connect()
```

Opens a connection to the database server. An InformixChannel sends this message to its InformixContext when the channel is about to open a connection to the server.

See Also: disconnect

connection

```
public int connection()
```

Returns an identifier for the receiver's connection to the server.

disconnect

```
public void disconnect()
```

Closes a connection to the database server. An InformixChannel sends this message to its InformixContext when the channel is about to close a connection to the server.

See Also: connect

CLASS InformixContext

fetchesInProgress

```
public int fetchesInProgress()
```

Returns the number of fetches the receiver has in progress.

hasTransactions

```
public boolean hasTransactions()
```

Returns `true` to indicate that the receiver has transactions in process, `false` otherwise.

isConnected

```
public boolean isConnected()
```

Returns `true` if the receiver has an open connection to the database, `false` otherwise.

See Also: `connect`, `disconnect`

isOnLine

```
public boolean isOnLine()
```

Returns `true` if the server is an Informix online server, `false` otherwise.

CLASS InformixContext

InformixSQLExpression

Inherits from: EOSQLExpression : NSObject

Package: com.apple.yellow.informixeadaptor

Class Description

InformixSQLExpression defines how to build SQL statements for InformixChannels.

Bind Variables

The InformixAdaptor uses bind variables. A bind variable is a placeholder used in an SQL statement that is replaced with an actual value after the database server determines an execution plan. You use the following methods to operate on bind variables:

- `bindValueDictionaryForAttribute`
- `mustUseBindVariableForAttribute`
- `shouldUseBindVariableForAttribute`

For more information on using bind variables, see the EOSQLExpression class specification.

Static Methods

serverTypeIdForName

```
public static int serverTypeIdForName(String typeName)
```

Returns the Informix type code (such as `InfDecimal`, `InfDate`, or `InfCHAR`) for *typeName* (such as "DECIMAL", "DATE", or "CHAR").

Instance Methods

lockClause

```
public String lockClause()
```

Overrides the `EOSQLExpression` method `lockClause` to return the SQL string used in a `SELECT` statement to lock selected rows, which is "FOR UPDATE OF".

InformixChannel.Delegate

(informal interface)

Package: com.apple.yellow.informixeadaptor

Interface Description

InformixChannel's delegate allows you to process errors that occur in the Informix server.

Instance Methods

informixChannelWillReportDatabaseError

```
public abstract boolean informixChannelWillReportDatabaseError(  
    InformixChannel channel,  
    String attributes)
```

Invoked whenever *channel* encounters an error reported by the Informix server. The *error* argument is the text of the Informix error message which will be sent to the adaptor's `reportError` method. The delegate can return `false` to prevent the channel from calling `reportError`.

INTERFACE InformixChannel.Delegate

This Apple manual was written, edited, and composed on a desktop publishing system using Apple Macintosh computers and FrameMaker software.

Line art was created using Adobe™ Illustrator and Adobe Photoshop.

Text type is Palatino® and display type is Helvetica®. Bullets are ITC Zapf Dingbats®. Some elements, such as program listings, are set in Adobe Letter Gothic.