| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/CallableStatement.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/sql/Blob.html)   [**NEXT CLASS**](http://docs.google.com/java/sql/ClientInfoStatus.html) | [**FRAMES**](http://docs.google.com/index.html?java/sql/CallableStatement.html)    [**NO FRAMES**](http://docs.google.com/CallableStatement.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#tyjcwt) | DETAIL: FIELD | CONSTR | [METHOD](#2s8eyo1) |

## **java.sql**

Interface CallableStatement

**All Superinterfaces:** [PreparedStatement](http://docs.google.com/java/sql/PreparedStatement.html), [Statement](http://docs.google.com/java/sql/Statement.html), [Wrapper](http://docs.google.com/java/sql/Wrapper.html)

public interface **CallableStatement**extends [PreparedStatement](http://docs.google.com/java/sql/PreparedStatement.html)

The interface used to execute SQL stored procedures. The JDBC API provides a stored procedure SQL escape syntax that allows stored procedures to be called in a standard way for all RDBMSs. This escape syntax has one form that includes a result parameter and one that does not. If used, the result parameter must be registered as an OUT parameter. The other parameters can be used for input, output or both. Parameters are referred to sequentially, by number, with the first parameter being 1.

{?= call <procedure-name>[(<arg1>,<arg2>, ...)]}  
 {call <procedure-name>[(<arg1>,<arg2>, ...)]}

IN parameter values are set using the set methods inherited from [PreparedStatement](http://docs.google.com/java/sql/PreparedStatement.html). The type of all OUT parameters must be registered prior to executing the stored procedure; their values are retrieved after execution via the get methods provided here.

A CallableStatement can return one [ResultSet](http://docs.google.com/java/sql/ResultSet.html) object or multiple ResultSet objects. Multiple ResultSet objects are handled using operations inherited from [Statement](http://docs.google.com/java/sql/Statement.html).

For maximum portability, a call's ResultSet objects and update counts should be processed prior to getting the values of output parameters.

**See Also:**[Connection.prepareCall(java.lang.String)](http://docs.google.com/java/sql/Connection.html#prepareCall(java.lang.String)), [ResultSet](http://docs.google.com/java/sql/ResultSet.html)

| **Field Summary** | |
| --- | --- |

| **Fields inherited from interface java.sql.**[**Statement**](http://docs.google.com/java/sql/Statement.html) |
| --- |
| [CLOSE\_ALL\_RESULTS](http://docs.google.com/java/sql/Statement.html#CLOSE_ALL_RESULTS), [CLOSE\_CURRENT\_RESULT](http://docs.google.com/java/sql/Statement.html#CLOSE_CURRENT_RESULT), [EXECUTE\_FAILED](http://docs.google.com/java/sql/Statement.html#EXECUTE_FAILED), [KEEP\_CURRENT\_RESULT](http://docs.google.com/java/sql/Statement.html#KEEP_CURRENT_RESULT), [NO\_GENERATED\_KEYS](http://docs.google.com/java/sql/Statement.html#NO_GENERATED_KEYS), [RETURN\_GENERATED\_KEYS](http://docs.google.com/java/sql/Statement.html#RETURN_GENERATED_KEYS), [SUCCESS\_NO\_INFO](http://docs.google.com/java/sql/Statement.html#SUCCESS_NO_INFO) |

| **Method Summary** | |
| --- | --- |
| [Array](http://docs.google.com/java/sql/Array.html) | [**getArray**](http://docs.google.com/java/sql/CallableStatement.html#getArray(int))(int parameterIndex)            Retrieves the value of the designated JDBC ARRAY parameter as an [Array](http://docs.google.com/java/sql/Array.html) object in the Java programming language. |
| [Array](http://docs.google.com/java/sql/Array.html) | [**getArray**](http://docs.google.com/java/sql/CallableStatement.html#getArray(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC ARRAY parameter as an [Array](http://docs.google.com/java/sql/Array.html) object in the Java programming language. |
| [BigDecimal](http://docs.google.com/java/math/BigDecimal.html) | [**getBigDecimal**](http://docs.google.com/java/sql/CallableStatement.html#getBigDecimal(int))(int parameterIndex)            Retrieves the value of the designated JDBC NUMERIC parameter as a java.math.BigDecimal object with as many digits to the right of the decimal point as the value contains. |
| [BigDecimal](http://docs.google.com/java/math/BigDecimal.html) | [**getBigDecimal**](http://docs.google.com/java/sql/CallableStatement.html#getBigDecimal(int,%20int))(int parameterIndex, int scale)  **Deprecated.** *use getBigDecimal(int parameterIndex) or getBigDecimal(String parameterName)* |
| [BigDecimal](http://docs.google.com/java/math/BigDecimal.html) | [**getBigDecimal**](http://docs.google.com/java/sql/CallableStatement.html#getBigDecimal(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC NUMERIC parameter as a java.math.BigDecimal object with as many digits to the right of the decimal point as the value contains. |
| [Blob](http://docs.google.com/java/sql/Blob.html) | [**getBlob**](http://docs.google.com/java/sql/CallableStatement.html#getBlob(int))(int parameterIndex)            Retrieves the value of the designated JDBC BLOB parameter as a [Blob](http://docs.google.com/java/sql/Blob.html) object in the Java programming language. |
| [Blob](http://docs.google.com/java/sql/Blob.html) | [**getBlob**](http://docs.google.com/java/sql/CallableStatement.html#getBlob(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC BLOB parameter as a [Blob](http://docs.google.com/java/sql/Blob.html) object in the Java programming language. |
| boolean | [**getBoolean**](http://docs.google.com/java/sql/CallableStatement.html#getBoolean(int))(int parameterIndex)            Retrieves the value of the designated JDBC BIT or BOOLEAN parameter as a boolean in the Java programming language. |
| boolean | [**getBoolean**](http://docs.google.com/java/sql/CallableStatement.html#getBoolean(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC BIT or BOOLEAN parameter as a boolean in the Java programming language. |
| byte | [**getByte**](http://docs.google.com/java/sql/CallableStatement.html#getByte(int))(int parameterIndex)            Retrieves the value of the designated JDBC TINYINT parameter as a byte in the Java programming language. |
| byte | [**getByte**](http://docs.google.com/java/sql/CallableStatement.html#getByte(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC TINYINT parameter as a byte in the Java programming language. |
| byte[] | [**getBytes**](http://docs.google.com/java/sql/CallableStatement.html#getBytes(int))(int parameterIndex)            Retrieves the value of the designated JDBC BINARY or VARBINARY parameter as an array of byte values in the Java programming language. |
| byte[] | [**getBytes**](http://docs.google.com/java/sql/CallableStatement.html#getBytes(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC BINARY or VARBINARY parameter as an array of byte values in the Java programming language. |
| [Reader](http://docs.google.com/java/io/Reader.html) | [**getCharacterStream**](http://docs.google.com/java/sql/CallableStatement.html#getCharacterStream(int))(int parameterIndex)            Retrieves the value of the designated parameter as a java.io.Reader object in the Java programming language. |
| [Reader](http://docs.google.com/java/io/Reader.html) | [**getCharacterStream**](http://docs.google.com/java/sql/CallableStatement.html#getCharacterStream(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of the designated parameter as a java.io.Reader object in the Java programming language. |
| [Clob](http://docs.google.com/java/sql/Clob.html) | [**getClob**](http://docs.google.com/java/sql/CallableStatement.html#getClob(int))(int parameterIndex)            Retrieves the value of the designated JDBC CLOB parameter as a java.sql.Clob object in the Java programming language. |
| [Clob](http://docs.google.com/java/sql/Clob.html) | [**getClob**](http://docs.google.com/java/sql/CallableStatement.html#getClob(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC CLOB parameter as a java.sql.Clob object in the Java programming language. |
| [Date](http://docs.google.com/java/sql/Date.html) | [**getDate**](http://docs.google.com/java/sql/CallableStatement.html#getDate(int))(int parameterIndex)            Retrieves the value of the designated JDBC DATE parameter as a java.sql.Date object. |
| [Date](http://docs.google.com/java/sql/Date.html) | [**getDate**](http://docs.google.com/java/sql/CallableStatement.html#getDate(int,%20java.util.Calendar))(int parameterIndex, [Calendar](http://docs.google.com/java/util/Calendar.html) cal)            Retrieves the value of the designated JDBC DATE parameter as a java.sql.Date object, using the given Calendar object to construct the date. |
| [Date](http://docs.google.com/java/sql/Date.html) | [**getDate**](http://docs.google.com/java/sql/CallableStatement.html#getDate(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC DATE parameter as a java.sql.Date object. |
| [Date](http://docs.google.com/java/sql/Date.html) | [**getDate**](http://docs.google.com/java/sql/CallableStatement.html#getDate(java.lang.String,%20java.util.Calendar))([String](http://docs.google.com/java/lang/String.html) parameterName, [Calendar](http://docs.google.com/java/util/Calendar.html) cal)            Retrieves the value of a JDBC DATE parameter as a java.sql.Date object, using the given Calendar object to construct the date. |
| double | [**getDouble**](http://docs.google.com/java/sql/CallableStatement.html#getDouble(int))(int parameterIndex)            Retrieves the value of the designated JDBC DOUBLE parameter as a double in the Java programming language. |
| double | [**getDouble**](http://docs.google.com/java/sql/CallableStatement.html#getDouble(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC DOUBLE parameter as a double in the Java programming language. |
| float | [**getFloat**](http://docs.google.com/java/sql/CallableStatement.html#getFloat(int))(int parameterIndex)            Retrieves the value of the designated JDBC FLOAT parameter as a float in the Java programming language. |
| float | [**getFloat**](http://docs.google.com/java/sql/CallableStatement.html#getFloat(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC FLOAT parameter as a float in the Java programming language. |
| int | [**getInt**](http://docs.google.com/java/sql/CallableStatement.html#getInt(int))(int parameterIndex)            Retrieves the value of the designated JDBC INTEGER parameter as an int in the Java programming language. |
| int | [**getInt**](http://docs.google.com/java/sql/CallableStatement.html#getInt(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC INTEGER parameter as an int in the Java programming language. |
| long | [**getLong**](http://docs.google.com/java/sql/CallableStatement.html#getLong(int))(int parameterIndex)            Retrieves the value of the designated JDBC BIGINT parameter as a long in the Java programming language. |
| long | [**getLong**](http://docs.google.com/java/sql/CallableStatement.html#getLong(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC BIGINT parameter as a long in the Java programming language. |
| [Reader](http://docs.google.com/java/io/Reader.html) | [**getNCharacterStream**](http://docs.google.com/java/sql/CallableStatement.html#getNCharacterStream(int))(int parameterIndex)            Retrieves the value of the designated parameter as a java.io.Reader object in the Java programming language. |
| [Reader](http://docs.google.com/java/io/Reader.html) | [**getNCharacterStream**](http://docs.google.com/java/sql/CallableStatement.html#getNCharacterStream(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of the designated parameter as a java.io.Reader object in the Java programming language. |
| [NClob](http://docs.google.com/java/sql/NClob.html) | [**getNClob**](http://docs.google.com/java/sql/CallableStatement.html#getNClob(int))(int parameterIndex)            Retrieves the value of the designated JDBC NCLOB parameter as a java.sql.NClob object in the Java programming language. |
| [NClob](http://docs.google.com/java/sql/NClob.html) | [**getNClob**](http://docs.google.com/java/sql/CallableStatement.html#getNClob(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC NCLOB parameter as a java.sql.NClob object in the Java programming language. |
| [String](http://docs.google.com/java/lang/String.html) | [**getNString**](http://docs.google.com/java/sql/CallableStatement.html#getNString(int))(int parameterIndex)            Retrieves the value of the designated NCHAR, NVARCHAR or LONGNVARCHAR parameter as a String in the Java programming language. |
| [String](http://docs.google.com/java/lang/String.html) | [**getNString**](http://docs.google.com/java/sql/CallableStatement.html#getNString(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of the designated NCHAR, NVARCHAR or LONGNVARCHAR parameter as a String in the Java programming language. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getObject**](http://docs.google.com/java/sql/CallableStatement.html#getObject(int))(int parameterIndex)            Retrieves the value of the designated parameter as an Object in the Java programming language. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getObject**](http://docs.google.com/java/sql/CallableStatement.html#getObject(int,%20java.util.Map))(int parameterIndex, [Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),[Class](http://docs.google.com/java/lang/Class.html)<?>> map)            Returns an object representing the value of OUT parameter parameterIndex and uses map for the custom mapping of the parameter value. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getObject**](http://docs.google.com/java/sql/CallableStatement.html#getObject(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a parameter as an Object in the Java programming language. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getObject**](http://docs.google.com/java/sql/CallableStatement.html#getObject(java.lang.String,%20java.util.Map))([String](http://docs.google.com/java/lang/String.html) parameterName, [Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),[Class](http://docs.google.com/java/lang/Class.html)<?>> map)            Returns an object representing the value of OUT parameter parameterName and uses map for the custom mapping of the parameter value. |
| [Ref](http://docs.google.com/java/sql/Ref.html) | [**getRef**](http://docs.google.com/java/sql/CallableStatement.html#getRef(int))(int parameterIndex)            Retrieves the value of the designated JDBC REF(<structured-type>) parameter as a [Ref](http://docs.google.com/java/sql/Ref.html) object in the Java programming language. |
| [Ref](http://docs.google.com/java/sql/Ref.html) | [**getRef**](http://docs.google.com/java/sql/CallableStatement.html#getRef(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC REF(<structured-type>) parameter as a [Ref](http://docs.google.com/java/sql/Ref.html) object in the Java programming language. |
| [RowId](http://docs.google.com/java/sql/RowId.html) | [**getRowId**](http://docs.google.com/java/sql/CallableStatement.html#getRowId(int))(int parameterIndex)            Retrieves the value of the designated JDBC ROWID parameter as a java.sql.RowId object. |
| [RowId](http://docs.google.com/java/sql/RowId.html) | [**getRowId**](http://docs.google.com/java/sql/CallableStatement.html#getRowId(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of the designated JDBC ROWID parameter as a java.sql.RowId object. |
| short | [**getShort**](http://docs.google.com/java/sql/CallableStatement.html#getShort(int))(int parameterIndex)            Retrieves the value of the designated JDBC SMALLINT parameter as a short in the Java programming language. |
| short | [**getShort**](http://docs.google.com/java/sql/CallableStatement.html#getShort(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC SMALLINT parameter as a short in the Java programming language. |
| [SQLXML](http://docs.google.com/java/sql/SQLXML.html) | [**getSQLXML**](http://docs.google.com/java/sql/CallableStatement.html#getSQLXML(int))(int parameterIndex)            Retrieves the value of the designated SQL XML parameter as a java.sql.SQLXML object in the Java programming language. |
| [SQLXML](http://docs.google.com/java/sql/SQLXML.html) | [**getSQLXML**](http://docs.google.com/java/sql/CallableStatement.html#getSQLXML(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of the designated SQL XML parameter as a java.sql.SQLXML object in the Java programming language. |
| [String](http://docs.google.com/java/lang/String.html) | [**getString**](http://docs.google.com/java/sql/CallableStatement.html#getString(int))(int parameterIndex)            Retrieves the value of the designated JDBC CHAR, VARCHAR, or LONGVARCHAR parameter as a String in the Java programming language. |
| [String](http://docs.google.com/java/lang/String.html) | [**getString**](http://docs.google.com/java/sql/CallableStatement.html#getString(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC CHAR, VARCHAR, or LONGVARCHAR parameter as a String in the Java programming language. |
| [Time](http://docs.google.com/java/sql/Time.html) | [**getTime**](http://docs.google.com/java/sql/CallableStatement.html#getTime(int))(int parameterIndex)            Retrieves the value of the designated JDBC TIME parameter as a java.sql.Time object. |
| [Time](http://docs.google.com/java/sql/Time.html) | [**getTime**](http://docs.google.com/java/sql/CallableStatement.html#getTime(int,%20java.util.Calendar))(int parameterIndex, [Calendar](http://docs.google.com/java/util/Calendar.html) cal)            Retrieves the value of the designated JDBC TIME parameter as a java.sql.Time object, using the given Calendar object to construct the time. |
| [Time](http://docs.google.com/java/sql/Time.html) | [**getTime**](http://docs.google.com/java/sql/CallableStatement.html#getTime(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC TIME parameter as a java.sql.Time object. |
| [Time](http://docs.google.com/java/sql/Time.html) | [**getTime**](http://docs.google.com/java/sql/CallableStatement.html#getTime(java.lang.String,%20java.util.Calendar))([String](http://docs.google.com/java/lang/String.html) parameterName, [Calendar](http://docs.google.com/java/util/Calendar.html) cal)            Retrieves the value of a JDBC TIME parameter as a java.sql.Time object, using the given Calendar object to construct the time. |
| [Timestamp](http://docs.google.com/java/sql/Timestamp.html) | [**getTimestamp**](http://docs.google.com/java/sql/CallableStatement.html#getTimestamp(int))(int parameterIndex)            Retrieves the value of the designated JDBC TIMESTAMP parameter as a java.sql.Timestamp object. |
| [Timestamp](http://docs.google.com/java/sql/Timestamp.html) | [**getTimestamp**](http://docs.google.com/java/sql/CallableStatement.html#getTimestamp(int,%20java.util.Calendar))(int parameterIndex, [Calendar](http://docs.google.com/java/util/Calendar.html) cal)            Retrieves the value of the designated JDBC TIMESTAMP parameter as a java.sql.Timestamp object, using the given Calendar object to construct the Timestamp object. |
| [Timestamp](http://docs.google.com/java/sql/Timestamp.html) | [**getTimestamp**](http://docs.google.com/java/sql/CallableStatement.html#getTimestamp(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC TIMESTAMP parameter as a java.sql.Timestamp object. |
| [Timestamp](http://docs.google.com/java/sql/Timestamp.html) | [**getTimestamp**](http://docs.google.com/java/sql/CallableStatement.html#getTimestamp(java.lang.String,%20java.util.Calendar))([String](http://docs.google.com/java/lang/String.html) parameterName, [Calendar](http://docs.google.com/java/util/Calendar.html) cal)            Retrieves the value of a JDBC TIMESTAMP parameter as a java.sql.Timestamp object, using the given Calendar object to construct the Timestamp object. |
| [URL](http://docs.google.com/java/net/URL.html) | [**getURL**](http://docs.google.com/java/sql/CallableStatement.html#getURL(int))(int parameterIndex)            Retrieves the value of the designated JDBC DATALINK parameter as a java.net.URL object. |
| [URL](http://docs.google.com/java/net/URL.html) | [**getURL**](http://docs.google.com/java/sql/CallableStatement.html#getURL(java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName)            Retrieves the value of a JDBC DATALINK parameter as a java.net.URL object. |
| void | [**registerOutParameter**](http://docs.google.com/java/sql/CallableStatement.html#registerOutParameter(int,%20int))(int parameterIndex, int sqlType)            Registers the OUT parameter in ordinal position parameterIndex to the JDBC type sqlType. |
| void | [**registerOutParameter**](http://docs.google.com/java/sql/CallableStatement.html#registerOutParameter(int,%20int,%20int))(int parameterIndex, int sqlType, int scale)            Registers the parameter in ordinal position parameterIndex to be of JDBC type sqlType. |
| void | [**registerOutParameter**](http://docs.google.com/java/sql/CallableStatement.html#registerOutParameter(int,%20int,%20java.lang.String))(int parameterIndex, int sqlType, [String](http://docs.google.com/java/lang/String.html) typeName)            Registers the designated output parameter. |
| void | [**registerOutParameter**](http://docs.google.com/java/sql/CallableStatement.html#registerOutParameter(java.lang.String,%20int))([String](http://docs.google.com/java/lang/String.html) parameterName, int sqlType)            Registers the OUT parameter named parameterName to the JDBC type sqlType. |
| void | [**registerOutParameter**](http://docs.google.com/java/sql/CallableStatement.html#registerOutParameter(java.lang.String,%20int,%20int))([String](http://docs.google.com/java/lang/String.html) parameterName, int sqlType, int scale)            Registers the parameter named parameterName to be of JDBC type sqlType. |
| void | [**registerOutParameter**](http://docs.google.com/java/sql/CallableStatement.html#registerOutParameter(java.lang.String,%20int,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName, int sqlType, [String](http://docs.google.com/java/lang/String.html) typeName)            Registers the designated output parameter. |
| void | [**setAsciiStream**](http://docs.google.com/java/sql/CallableStatement.html#setAsciiStream(java.lang.String,%20java.io.InputStream))([String](http://docs.google.com/java/lang/String.html) parameterName, [InputStream](http://docs.google.com/java/io/InputStream.html) x)            Sets the designated parameter to the given input stream. |
| void | [**setAsciiStream**](http://docs.google.com/java/sql/CallableStatement.html#setAsciiStream(java.lang.String,%20java.io.InputStream,%20int))([String](http://docs.google.com/java/lang/String.html) parameterName, [InputStream](http://docs.google.com/java/io/InputStream.html) x, int length)            Sets the designated parameter to the given input stream, which will have the specified number of bytes. |
| void | [**setAsciiStream**](http://docs.google.com/java/sql/CallableStatement.html#setAsciiStream(java.lang.String,%20java.io.InputStream,%20long))([String](http://docs.google.com/java/lang/String.html) parameterName, [InputStream](http://docs.google.com/java/io/InputStream.html) x, long length)            Sets the designated parameter to the given input stream, which will have the specified number of bytes. |
| void | [**setBigDecimal**](http://docs.google.com/java/sql/CallableStatement.html#setBigDecimal(java.lang.String,%20java.math.BigDecimal))([String](http://docs.google.com/java/lang/String.html) parameterName, [BigDecimal](http://docs.google.com/java/math/BigDecimal.html) x)            Sets the designated parameter to the given java.math.BigDecimal value. |
| void | [**setBinaryStream**](http://docs.google.com/java/sql/CallableStatement.html#setBinaryStream(java.lang.String,%20java.io.InputStream))([String](http://docs.google.com/java/lang/String.html) parameterName, [InputStream](http://docs.google.com/java/io/InputStream.html) x)            Sets the designated parameter to the given input stream. |
| void | [**setBinaryStream**](http://docs.google.com/java/sql/CallableStatement.html#setBinaryStream(java.lang.String,%20java.io.InputStream,%20int))([String](http://docs.google.com/java/lang/String.html) parameterName, [InputStream](http://docs.google.com/java/io/InputStream.html) x, int length)            Sets the designated parameter to the given input stream, which will have the specified number of bytes. |
| void | [**setBinaryStream**](http://docs.google.com/java/sql/CallableStatement.html#setBinaryStream(java.lang.String,%20java.io.InputStream,%20long))([String](http://docs.google.com/java/lang/String.html) parameterName, [InputStream](http://docs.google.com/java/io/InputStream.html) x, long length)            Sets the designated parameter to the given input stream, which will have the specified number of bytes. |
| void | [**setBlob**](http://docs.google.com/java/sql/CallableStatement.html#setBlob(java.lang.String,%20java.sql.Blob))([String](http://docs.google.com/java/lang/String.html) parameterName, [Blob](http://docs.google.com/java/sql/Blob.html) x)            Sets the designated parameter to the given java.sql.Blob object. |
| void | [**setBlob**](http://docs.google.com/java/sql/CallableStatement.html#setBlob(java.lang.String,%20java.io.InputStream))([String](http://docs.google.com/java/lang/String.html) parameterName, [InputStream](http://docs.google.com/java/io/InputStream.html) inputStream)            Sets the designated parameter to a InputStream object. |
| void | [**setBlob**](http://docs.google.com/java/sql/CallableStatement.html#setBlob(java.lang.String,%20java.io.InputStream,%20long))([String](http://docs.google.com/java/lang/String.html) parameterName, [InputStream](http://docs.google.com/java/io/InputStream.html) inputStream, long length)            Sets the designated parameter to a InputStream object. |
| void | [**setBoolean**](http://docs.google.com/java/sql/CallableStatement.html#setBoolean(java.lang.String,%20boolean))([String](http://docs.google.com/java/lang/String.html) parameterName, boolean x)            Sets the designated parameter to the given Java boolean value. |
| void | [**setByte**](http://docs.google.com/java/sql/CallableStatement.html#setByte(java.lang.String,%20byte))([String](http://docs.google.com/java/lang/String.html) parameterName, byte x)            Sets the designated parameter to the given Java byte value. |
| void | [**setBytes**](http://docs.google.com/java/sql/CallableStatement.html#setBytes(java.lang.String,%20byte%5B%5D))([String](http://docs.google.com/java/lang/String.html) parameterName, byte[] x)            Sets the designated parameter to the given Java array of bytes. |
| void | [**setCharacterStream**](http://docs.google.com/java/sql/CallableStatement.html#setCharacterStream(java.lang.String,%20java.io.Reader))([String](http://docs.google.com/java/lang/String.html) parameterName, [Reader](http://docs.google.com/java/io/Reader.html) reader)            Sets the designated parameter to the given Reader object. |
| void | [**setCharacterStream**](http://docs.google.com/java/sql/CallableStatement.html#setCharacterStream(java.lang.String,%20java.io.Reader,%20int))([String](http://docs.google.com/java/lang/String.html) parameterName, [Reader](http://docs.google.com/java/io/Reader.html) reader, int length)            Sets the designated parameter to the given Reader object, which is the given number of characters long. |
| void | [**setCharacterStream**](http://docs.google.com/java/sql/CallableStatement.html#setCharacterStream(java.lang.String,%20java.io.Reader,%20long))([String](http://docs.google.com/java/lang/String.html) parameterName, [Reader](http://docs.google.com/java/io/Reader.html) reader, long length)            Sets the designated parameter to the given Reader object, which is the given number of characters long. |
| void | [**setClob**](http://docs.google.com/java/sql/CallableStatement.html#setClob(java.lang.String,%20java.sql.Clob))([String](http://docs.google.com/java/lang/String.html) parameterName, [Clob](http://docs.google.com/java/sql/Clob.html) x)            Sets the designated parameter to the given java.sql.Clob object. |
| void | [**setClob**](http://docs.google.com/java/sql/CallableStatement.html#setClob(java.lang.String,%20java.io.Reader))([String](http://docs.google.com/java/lang/String.html) parameterName, [Reader](http://docs.google.com/java/io/Reader.html) reader)            Sets the designated parameter to a Reader object. |
| void | [**setClob**](http://docs.google.com/java/sql/CallableStatement.html#setClob(java.lang.String,%20java.io.Reader,%20long))([String](http://docs.google.com/java/lang/String.html) parameterName, [Reader](http://docs.google.com/java/io/Reader.html) reader, long length)            Sets the designated parameter to a Reader object. |
| void | [**setDate**](http://docs.google.com/java/sql/CallableStatement.html#setDate(java.lang.String,%20java.sql.Date))([String](http://docs.google.com/java/lang/String.html) parameterName, [Date](http://docs.google.com/java/sql/Date.html) x)            Sets the designated parameter to the given java.sql.Date value using the default time zone of the virtual machine that is running the application. |
| void | [**setDate**](http://docs.google.com/java/sql/CallableStatement.html#setDate(java.lang.String,%20java.sql.Date,%20java.util.Calendar))([String](http://docs.google.com/java/lang/String.html) parameterName, [Date](http://docs.google.com/java/sql/Date.html) x, [Calendar](http://docs.google.com/java/util/Calendar.html) cal)            Sets the designated parameter to the given java.sql.Date value, using the given Calendar object. |
| void | [**setDouble**](http://docs.google.com/java/sql/CallableStatement.html#setDouble(java.lang.String,%20double))([String](http://docs.google.com/java/lang/String.html) parameterName, double x)            Sets the designated parameter to the given Java double value. |
| void | [**setFloat**](http://docs.google.com/java/sql/CallableStatement.html#setFloat(java.lang.String,%20float))([String](http://docs.google.com/java/lang/String.html) parameterName, float x)            Sets the designated parameter to the given Java float value. |
| void | [**setInt**](http://docs.google.com/java/sql/CallableStatement.html#setInt(java.lang.String,%20int))([String](http://docs.google.com/java/lang/String.html) parameterName, int x)            Sets the designated parameter to the given Java int value. |
| void | [**setLong**](http://docs.google.com/java/sql/CallableStatement.html#setLong(java.lang.String,%20long))([String](http://docs.google.com/java/lang/String.html) parameterName, long x)            Sets the designated parameter to the given Java long value. |
| void | [**setNCharacterStream**](http://docs.google.com/java/sql/CallableStatement.html#setNCharacterStream(java.lang.String,%20java.io.Reader))([String](http://docs.google.com/java/lang/String.html) parameterName, [Reader](http://docs.google.com/java/io/Reader.html) value)            Sets the designated parameter to a Reader object. |
| void | [**setNCharacterStream**](http://docs.google.com/java/sql/CallableStatement.html#setNCharacterStream(java.lang.String,%20java.io.Reader,%20long))([String](http://docs.google.com/java/lang/String.html) parameterName, [Reader](http://docs.google.com/java/io/Reader.html) value, long length)            Sets the designated parameter to a Reader object. |
| void | [**setNClob**](http://docs.google.com/java/sql/CallableStatement.html#setNClob(java.lang.String,%20java.sql.NClob))([String](http://docs.google.com/java/lang/String.html) parameterName, [NClob](http://docs.google.com/java/sql/NClob.html) value)            Sets the designated parameter to a java.sql.NClob object. |
| void | [**setNClob**](http://docs.google.com/java/sql/CallableStatement.html#setNClob(java.lang.String,%20java.io.Reader))([String](http://docs.google.com/java/lang/String.html) parameterName, [Reader](http://docs.google.com/java/io/Reader.html) reader)            Sets the designated parameter to a Reader object. |
| void | [**setNClob**](http://docs.google.com/java/sql/CallableStatement.html#setNClob(java.lang.String,%20java.io.Reader,%20long))([String](http://docs.google.com/java/lang/String.html) parameterName, [Reader](http://docs.google.com/java/io/Reader.html) reader, long length)            Sets the designated parameter to a Reader object. |
| void | [**setNString**](http://docs.google.com/java/sql/CallableStatement.html#setNString(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName, [String](http://docs.google.com/java/lang/String.html) value)            Sets the designated parameter to the given String object. |
| void | [**setNull**](http://docs.google.com/java/sql/CallableStatement.html#setNull(java.lang.String,%20int))([String](http://docs.google.com/java/lang/String.html) parameterName, int sqlType)            Sets the designated parameter to SQL NULL. |
| void | [**setNull**](http://docs.google.com/java/sql/CallableStatement.html#setNull(java.lang.String,%20int,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName, int sqlType, [String](http://docs.google.com/java/lang/String.html) typeName)            Sets the designated parameter to SQL NULL. |
| void | [**setObject**](http://docs.google.com/java/sql/CallableStatement.html#setObject(java.lang.String,%20java.lang.Object))([String](http://docs.google.com/java/lang/String.html) parameterName, [Object](http://docs.google.com/java/lang/Object.html) x)            Sets the value of the designated parameter with the given object. |
| void | [**setObject**](http://docs.google.com/java/sql/CallableStatement.html#setObject(java.lang.String,%20java.lang.Object,%20int))([String](http://docs.google.com/java/lang/String.html) parameterName, [Object](http://docs.google.com/java/lang/Object.html) x, int targetSqlType)            Sets the value of the designated parameter with the given object. |
| void | [**setObject**](http://docs.google.com/java/sql/CallableStatement.html#setObject(java.lang.String,%20java.lang.Object,%20int,%20int))([String](http://docs.google.com/java/lang/String.html) parameterName, [Object](http://docs.google.com/java/lang/Object.html) x, int targetSqlType, int scale)            Sets the value of the designated parameter with the given object. |
| void | [**setRowId**](http://docs.google.com/java/sql/CallableStatement.html#setRowId(java.lang.String,%20java.sql.RowId))([String](http://docs.google.com/java/lang/String.html) parameterName, [RowId](http://docs.google.com/java/sql/RowId.html) x)            Sets the designated parameter to the given java.sql.RowId object. |
| void | [**setShort**](http://docs.google.com/java/sql/CallableStatement.html#setShort(java.lang.String,%20short))([String](http://docs.google.com/java/lang/String.html) parameterName, short x)            Sets the designated parameter to the given Java short value. |
| void | [**setSQLXML**](http://docs.google.com/java/sql/CallableStatement.html#setSQLXML(java.lang.String,%20java.sql.SQLXML))([String](http://docs.google.com/java/lang/String.html) parameterName, [SQLXML](http://docs.google.com/java/sql/SQLXML.html) xmlObject)            Sets the designated parameter to the given java.sql.SQLXML object. |
| void | [**setString**](http://docs.google.com/java/sql/CallableStatement.html#setString(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName, [String](http://docs.google.com/java/lang/String.html) x)            Sets the designated parameter to the given Java String value. |
| void | [**setTime**](http://docs.google.com/java/sql/CallableStatement.html#setTime(java.lang.String,%20java.sql.Time))([String](http://docs.google.com/java/lang/String.html) parameterName, [Time](http://docs.google.com/java/sql/Time.html) x)            Sets the designated parameter to the given java.sql.Time value. |
| void | [**setTime**](http://docs.google.com/java/sql/CallableStatement.html#setTime(java.lang.String,%20java.sql.Time,%20java.util.Calendar))([String](http://docs.google.com/java/lang/String.html) parameterName, [Time](http://docs.google.com/java/sql/Time.html) x, [Calendar](http://docs.google.com/java/util/Calendar.html) cal)            Sets the designated parameter to the given java.sql.Time value, using the given Calendar object. |
| void | [**setTimestamp**](http://docs.google.com/java/sql/CallableStatement.html#setTimestamp(java.lang.String,%20java.sql.Timestamp))([String](http://docs.google.com/java/lang/String.html) parameterName, [Timestamp](http://docs.google.com/java/sql/Timestamp.html) x)            Sets the designated parameter to the given java.sql.Timestamp value. |
| void | [**setTimestamp**](http://docs.google.com/java/sql/CallableStatement.html#setTimestamp(java.lang.String,%20java.sql.Timestamp,%20java.util.Calendar))([String](http://docs.google.com/java/lang/String.html) parameterName, [Timestamp](http://docs.google.com/java/sql/Timestamp.html) x, [Calendar](http://docs.google.com/java/util/Calendar.html) cal)            Sets the designated parameter to the given java.sql.Timestamp value, using the given Calendar object. |
| void | [**setURL**](http://docs.google.com/java/sql/CallableStatement.html#setURL(java.lang.String,%20java.net.URL))([String](http://docs.google.com/java/lang/String.html) parameterName, [URL](http://docs.google.com/java/net/URL.html) val)            Sets the designated parameter to the given java.net.URL object. |
| boolean | [**wasNull**](http://docs.google.com/java/sql/CallableStatement.html#wasNull())()            Retrieves whether the last OUT parameter read had the value of SQL NULL. |

| **Methods inherited from interface java.sql.**[**PreparedStatement**](http://docs.google.com/java/sql/PreparedStatement.html) |
| --- |
| [addBatch](http://docs.google.com/java/sql/PreparedStatement.html#addBatch()), [clearParameters](http://docs.google.com/java/sql/PreparedStatement.html#clearParameters()), [execute](http://docs.google.com/java/sql/PreparedStatement.html#execute()), [executeQuery](http://docs.google.com/java/sql/PreparedStatement.html#executeQuery()), [executeUpdate](http://docs.google.com/java/sql/PreparedStatement.html#executeUpdate()), [getMetaData](http://docs.google.com/java/sql/PreparedStatement.html#getMetaData()), [getParameterMetaData](http://docs.google.com/java/sql/PreparedStatement.html#getParameterMetaData()), [setArray](http://docs.google.com/java/sql/PreparedStatement.html#setArray(int,%20java.sql.Array)), [setAsciiStream](http://docs.google.com/java/sql/PreparedStatement.html#setAsciiStream(int,%20java.io.InputStream)), [setAsciiStream](http://docs.google.com/java/sql/PreparedStatement.html#setAsciiStream(int,%20java.io.InputStream,%20int)), [setAsciiStream](http://docs.google.com/java/sql/PreparedStatement.html#setAsciiStream(int,%20java.io.InputStream,%20long)), [setBigDecimal](http://docs.google.com/java/sql/PreparedStatement.html#setBigDecimal(int,%20java.math.BigDecimal)), [setBinaryStream](http://docs.google.com/java/sql/PreparedStatement.html#setBinaryStream(int,%20java.io.InputStream)), [setBinaryStream](http://docs.google.com/java/sql/PreparedStatement.html#setBinaryStream(int,%20java.io.InputStream,%20int)), [setBinaryStream](http://docs.google.com/java/sql/PreparedStatement.html#setBinaryStream(int,%20java.io.InputStream,%20long)), [setBlob](http://docs.google.com/java/sql/PreparedStatement.html#setBlob(int,%20java.sql.Blob)), [setBlob](http://docs.google.com/java/sql/PreparedStatement.html#setBlob(int,%20java.io.InputStream)), [setBlob](http://docs.google.com/java/sql/PreparedStatement.html#setBlob(int,%20java.io.InputStream,%20long)), [setBoolean](http://docs.google.com/java/sql/PreparedStatement.html#setBoolean(int,%20boolean)), [setByte](http://docs.google.com/java/sql/PreparedStatement.html#setByte(int,%20byte)), [setBytes](http://docs.google.com/java/sql/PreparedStatement.html#setBytes(int,%20byte%5B%5D)), [setCharacterStream](http://docs.google.com/java/sql/PreparedStatement.html#setCharacterStream(int,%20java.io.Reader)), [setCharacterStream](http://docs.google.com/java/sql/PreparedStatement.html#setCharacterStream(int,%20java.io.Reader,%20int)), [setCharacterStream](http://docs.google.com/java/sql/PreparedStatement.html#setCharacterStream(int,%20java.io.Reader,%20long)), [setClob](http://docs.google.com/java/sql/PreparedStatement.html#setClob(int,%20java.sql.Clob)), [setClob](http://docs.google.com/java/sql/PreparedStatement.html#setClob(int,%20java.io.Reader)), [setClob](http://docs.google.com/java/sql/PreparedStatement.html#setClob(int,%20java.io.Reader,%20long)), [setDate](http://docs.google.com/java/sql/PreparedStatement.html#setDate(int,%20java.sql.Date)), [setDate](http://docs.google.com/java/sql/PreparedStatement.html#setDate(int,%20java.sql.Date,%20java.util.Calendar)), [setDouble](http://docs.google.com/java/sql/PreparedStatement.html#setDouble(int,%20double)), [setFloat](http://docs.google.com/java/sql/PreparedStatement.html#setFloat(int,%20float)), [setInt](http://docs.google.com/java/sql/PreparedStatement.html#setInt(int,%20int)), [setLong](http://docs.google.com/java/sql/PreparedStatement.html#setLong(int,%20long)), [setNCharacterStream](http://docs.google.com/java/sql/PreparedStatement.html#setNCharacterStream(int,%20java.io.Reader)), [setNCharacterStream](http://docs.google.com/java/sql/PreparedStatement.html#setNCharacterStream(int,%20java.io.Reader,%20long)), [setNClob](http://docs.google.com/java/sql/PreparedStatement.html#setNClob(int,%20java.sql.NClob)), [setNClob](http://docs.google.com/java/sql/PreparedStatement.html#setNClob(int,%20java.io.Reader)), [setNClob](http://docs.google.com/java/sql/PreparedStatement.html#setNClob(int,%20java.io.Reader,%20long)), [setNString](http://docs.google.com/java/sql/PreparedStatement.html#setNString(int,%20java.lang.String)), [setNull](http://docs.google.com/java/sql/PreparedStatement.html#setNull(int,%20int)), [setNull](http://docs.google.com/java/sql/PreparedStatement.html#setNull(int,%20int,%20java.lang.String)), [setObject](http://docs.google.com/java/sql/PreparedStatement.html#setObject(int,%20java.lang.Object)), [setObject](http://docs.google.com/java/sql/PreparedStatement.html#setObject(int,%20java.lang.Object,%20int)), [setObject](http://docs.google.com/java/sql/PreparedStatement.html#setObject(int,%20java.lang.Object,%20int,%20int)), [setRef](http://docs.google.com/java/sql/PreparedStatement.html#setRef(int,%20java.sql.Ref)), [setRowId](http://docs.google.com/java/sql/PreparedStatement.html#setRowId(int,%20java.sql.RowId)), [setShort](http://docs.google.com/java/sql/PreparedStatement.html#setShort(int,%20short)), [setSQLXML](http://docs.google.com/java/sql/PreparedStatement.html#setSQLXML(int,%20java.sql.SQLXML)), [setString](http://docs.google.com/java/sql/PreparedStatement.html#setString(int,%20java.lang.String)), [setTime](http://docs.google.com/java/sql/PreparedStatement.html#setTime(int,%20java.sql.Time)), [setTime](http://docs.google.com/java/sql/PreparedStatement.html#setTime(int,%20java.sql.Time,%20java.util.Calendar)), [setTimestamp](http://docs.google.com/java/sql/PreparedStatement.html#setTimestamp(int,%20java.sql.Timestamp)), [setTimestamp](http://docs.google.com/java/sql/PreparedStatement.html#setTimestamp(int,%20java.sql.Timestamp,%20java.util.Calendar)), [setUnicodeStream](http://docs.google.com/java/sql/PreparedStatement.html#setUnicodeStream(int,%20java.io.InputStream,%20int)), [setURL](http://docs.google.com/java/sql/PreparedStatement.html#setURL(int,%20java.net.URL)) |

| **Methods inherited from interface java.sql.**[**Statement**](http://docs.google.com/java/sql/Statement.html) |
| --- |
| [addBatch](http://docs.google.com/java/sql/Statement.html#addBatch(java.lang.String)), [cancel](http://docs.google.com/java/sql/Statement.html#cancel()), [clearBatch](http://docs.google.com/java/sql/Statement.html#clearBatch()), [clearWarnings](http://docs.google.com/java/sql/Statement.html#clearWarnings()), [close](http://docs.google.com/java/sql/Statement.html#close()), [execute](http://docs.google.com/java/sql/Statement.html#execute(java.lang.String)), [execute](http://docs.google.com/java/sql/Statement.html#execute(java.lang.String,%20int)), [execute](http://docs.google.com/java/sql/Statement.html#execute(java.lang.String,%20int%5B%5D)), [execute](http://docs.google.com/java/sql/Statement.html#execute(java.lang.String,%20java.lang.String%5B%5D)), [executeBatch](http://docs.google.com/java/sql/Statement.html#executeBatch()), [executeQuery](http://docs.google.com/java/sql/Statement.html#executeQuery(java.lang.String)), [executeUpdate](http://docs.google.com/java/sql/Statement.html#executeUpdate(java.lang.String)), [executeUpdate](http://docs.google.com/java/sql/Statement.html#executeUpdate(java.lang.String,%20int)), [executeUpdate](http://docs.google.com/java/sql/Statement.html#executeUpdate(java.lang.String,%20int%5B%5D)), [executeUpdate](http://docs.google.com/java/sql/Statement.html#executeUpdate(java.lang.String,%20java.lang.String%5B%5D)), [getConnection](http://docs.google.com/java/sql/Statement.html#getConnection()), [getFetchDirection](http://docs.google.com/java/sql/Statement.html#getFetchDirection()), [getFetchSize](http://docs.google.com/java/sql/Statement.html#getFetchSize()), [getGeneratedKeys](http://docs.google.com/java/sql/Statement.html#getGeneratedKeys()), [getMaxFieldSize](http://docs.google.com/java/sql/Statement.html#getMaxFieldSize()), [getMaxRows](http://docs.google.com/java/sql/Statement.html#getMaxRows()), [getMoreResults](http://docs.google.com/java/sql/Statement.html#getMoreResults()), [getMoreResults](http://docs.google.com/java/sql/Statement.html#getMoreResults(int)), [getQueryTimeout](http://docs.google.com/java/sql/Statement.html#getQueryTimeout()), [getResultSet](http://docs.google.com/java/sql/Statement.html#getResultSet()), [getResultSetConcurrency](http://docs.google.com/java/sql/Statement.html#getResultSetConcurrency()), [getResultSetHoldability](http://docs.google.com/java/sql/Statement.html#getResultSetHoldability()), [getResultSetType](http://docs.google.com/java/sql/Statement.html#getResultSetType()), [getUpdateCount](http://docs.google.com/java/sql/Statement.html#getUpdateCount()), [getWarnings](http://docs.google.com/java/sql/Statement.html#getWarnings()), [isClosed](http://docs.google.com/java/sql/Statement.html#isClosed()), [isPoolable](http://docs.google.com/java/sql/Statement.html#isPoolable()), [setCursorName](http://docs.google.com/java/sql/Statement.html#setCursorName(java.lang.String)), [setEscapeProcessing](http://docs.google.com/java/sql/Statement.html#setEscapeProcessing(boolean)), [setFetchDirection](http://docs.google.com/java/sql/Statement.html#setFetchDirection(int)), [setFetchSize](http://docs.google.com/java/sql/Statement.html#setFetchSize(int)), [setMaxFieldSize](http://docs.google.com/java/sql/Statement.html#setMaxFieldSize(int)), [setMaxRows](http://docs.google.com/java/sql/Statement.html#setMaxRows(int)), [setPoolable](http://docs.google.com/java/sql/Statement.html#setPoolable(boolean)), [setQueryTimeout](http://docs.google.com/java/sql/Statement.html#setQueryTimeout(int)) |

| **Methods inherited from interface java.sql.**[**Wrapper**](http://docs.google.com/java/sql/Wrapper.html) |
| --- |
| [isWrapperFor](http://docs.google.com/java/sql/Wrapper.html#isWrapperFor(java.lang.Class)), [unwrap](http://docs.google.com/java/sql/Wrapper.html#unwrap(java.lang.Class)) |

| **Method Detail** |
| --- |

### registerOutParameter

void **registerOutParameter**(int parameterIndex,  
 int sqlType)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Registers the OUT parameter in ordinal position parameterIndex to the JDBC type sqlType. All OUT parameters must be registered before a stored procedure is executed.

The JDBC type specified by sqlType for an OUT parameter determines the Java type that must be used in the get method to read the value of that parameter.

If the JDBC type expected to be returned to this output parameter is specific to this particular database, sqlType should be java.sql.Types.OTHER. The method [getObject(int)](http://docs.google.com/java/sql/CallableStatement.html#getObject(int)) retrieves the value.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so onsqlType - the JDBC type code defined by java.sql.Types. If the parameter is of JDBC type NUMERIC or DECIMAL, the version of registerOutParameter that accepts a scale value should be used. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if sqlType is a ARRAY, BLOB, CLOB, DATALINK, JAVA\_OBJECT, NCHAR, NCLOB, NVARCHAR, LONGNVARCHAR, REF, ROWID, SQLXML or STRUCT data type and the JDBC driver does not support this data type**See Also:**[Types](http://docs.google.com/java/sql/Types.html)

### registerOutParameter

void **registerOutParameter**(int parameterIndex,  
 int sqlType,  
 int scale)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Registers the parameter in ordinal position parameterIndex to be of JDBC type sqlType. All OUT parameters must be registered before a stored procedure is executed.

The JDBC type specified by sqlType for an OUT parameter determines the Java type that must be used in the get method to read the value of that parameter.

This version of registerOutParameter should be used when the parameter is of JDBC type NUMERIC or DECIMAL.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so onsqlType - the SQL type code defined by java.sql.Types.scale - the desired number of digits to the right of the decimal point. It must be greater than or equal to zero. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if sqlType is a ARRAY, BLOB, CLOB, DATALINK, JAVA\_OBJECT, NCHAR, NCLOB, NVARCHAR, LONGNVARCHAR, REF, ROWID, SQLXML or STRUCT data type and the JDBC driver does not support this data type**See Also:**[Types](http://docs.google.com/java/sql/Types.html)

### wasNull

boolean **wasNull**()  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves whether the last OUT parameter read had the value of SQL NULL. Note that this method should be called only after calling a getter method; otherwise, there is no value to use in determining whether it is null or not.

**Returns:**true if the last parameter read was SQL NULL; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs or this method is called on a closed CallableStatement

### getString

[String](http://docs.google.com/java/lang/String.html) **getString**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC CHAR, VARCHAR, or LONGVARCHAR parameter as a String in the Java programming language.

For the fixed-length type JDBC CHAR, the String object returned has exactly the same value the SQL CHAR value had in the database, including any padding added by the database.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**See Also:**[setString(java.lang.String, java.lang.String)](http://docs.google.com/java/sql/CallableStatement.html#setString(java.lang.String,%20java.lang.String))

### getBoolean

boolean **getBoolean**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC BIT or BOOLEAN parameter as a boolean in the Java programming language.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value. If the value is SQL NULL, the result is false. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**See Also:**[setBoolean(java.lang.String, boolean)](http://docs.google.com/java/sql/CallableStatement.html#setBoolean(java.lang.String,%20boolean))

### getByte

byte **getByte**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC TINYINT parameter as a byte in the Java programming language.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value. If the value is SQL NULL, the result is 0. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**See Also:**[setByte(java.lang.String, byte)](http://docs.google.com/java/sql/CallableStatement.html#setByte(java.lang.String,%20byte))

### getShort

short **getShort**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC SMALLINT parameter as a short in the Java programming language.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value. If the value is SQL NULL, the result is 0. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**See Also:**[setShort(java.lang.String, short)](http://docs.google.com/java/sql/CallableStatement.html#setShort(java.lang.String,%20short))

### getInt

int **getInt**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC INTEGER parameter as an int in the Java programming language.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value. If the value is SQL NULL, the result is 0. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**See Also:**[setInt(java.lang.String, int)](http://docs.google.com/java/sql/CallableStatement.html#setInt(java.lang.String,%20int))

### getLong

long **getLong**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC BIGINT parameter as a long in the Java programming language.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value. If the value is SQL NULL, the result is 0. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**See Also:**[setLong(java.lang.String, long)](http://docs.google.com/java/sql/CallableStatement.html#setLong(java.lang.String,%20long))

### getFloat

float **getFloat**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC FLOAT parameter as a float in the Java programming language.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value. If the value is SQL NULL, the result is 0. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**See Also:**[setFloat(java.lang.String, float)](http://docs.google.com/java/sql/CallableStatement.html#setFloat(java.lang.String,%20float))

### getDouble

double **getDouble**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC DOUBLE parameter as a double in the Java programming language.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value. If the value is SQL NULL, the result is 0. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**See Also:**[setDouble(java.lang.String, double)](http://docs.google.com/java/sql/CallableStatement.html#setDouble(java.lang.String,%20double))

### getBigDecimal

[BigDecimal](http://docs.google.com/java/math/BigDecimal.html) **getBigDecimal**(int parameterIndex,  
 int scale)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

**Deprecated.** *use getBigDecimal(int parameterIndex) or getBigDecimal(String parameterName)*

Retrieves the value of the designated JDBC NUMERIC parameter as a java.math.BigDecimal object with *scale* digits to the right of the decimal point.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so onscale - the number of digits to the right of the decimal point **Returns:**the parameter value. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**See Also:**[setBigDecimal(java.lang.String, java.math.BigDecimal)](http://docs.google.com/java/sql/CallableStatement.html#setBigDecimal(java.lang.String,%20java.math.BigDecimal))

### getBytes

byte[] **getBytes**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC BINARY or VARBINARY parameter as an array of byte values in the Java programming language.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**See Also:**[setBytes(java.lang.String, byte[])](http://docs.google.com/java/sql/CallableStatement.html#setBytes(java.lang.String,%20byte%5B%5D))

### getDate

[Date](http://docs.google.com/java/sql/Date.html) **getDate**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC DATE parameter as a java.sql.Date object.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**See Also:**[setDate(java.lang.String, java.sql.Date)](http://docs.google.com/java/sql/CallableStatement.html#setDate(java.lang.String,%20java.sql.Date))

### getTime

[Time](http://docs.google.com/java/sql/Time.html) **getTime**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC TIME parameter as a java.sql.Time object.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**See Also:**[setTime(java.lang.String, java.sql.Time)](http://docs.google.com/java/sql/CallableStatement.html#setTime(java.lang.String,%20java.sql.Time))

### getTimestamp

[Timestamp](http://docs.google.com/java/sql/Timestamp.html) **getTimestamp**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC TIMESTAMP parameter as a java.sql.Timestamp object.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**See Also:**[setTimestamp(java.lang.String, java.sql.Timestamp)](http://docs.google.com/java/sql/CallableStatement.html#setTimestamp(java.lang.String,%20java.sql.Timestamp))

### getObject

[Object](http://docs.google.com/java/lang/Object.html) **getObject**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated parameter as an Object in the Java programming language. If the value is an SQL NULL, the driver returns a Java null.

This method returns a Java object whose type corresponds to the JDBC type that was registered for this parameter using the method registerOutParameter. By registering the target JDBC type as java.sql.Types.OTHER, this method can be used to read database-specific abstract data types.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**A java.lang.Object holding the OUT parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**See Also:**[Types](http://docs.google.com/java/sql/Types.html), [setObject(java.lang.String, java.lang.Object, int, int)](http://docs.google.com/java/sql/CallableStatement.html#setObject(java.lang.String,%20java.lang.Object,%20int,%20int))

### getBigDecimal

[BigDecimal](http://docs.google.com/java/math/BigDecimal.html) **getBigDecimal**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC NUMERIC parameter as a java.math.BigDecimal object with as many digits to the right of the decimal point as the value contains.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value in full precision. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**Since:** 1.2 **See Also:**[setBigDecimal(java.lang.String, java.math.BigDecimal)](http://docs.google.com/java/sql/CallableStatement.html#setBigDecimal(java.lang.String,%20java.math.BigDecimal))

### getObject

[Object](http://docs.google.com/java/lang/Object.html) **getObject**(int parameterIndex,  
 [Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),[Class](http://docs.google.com/java/lang/Class.html)<?>> map)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Returns an object representing the value of OUT parameter parameterIndex and uses map for the custom mapping of the parameter value.

This method returns a Java object whose type corresponds to the JDBC type that was registered for this parameter using the method registerOutParameter. By registering the target JDBC type as java.sql.Types.OTHER, this method can be used to read database-specific abstract data types.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so onmap - the mapping from SQL type names to Java classes **Returns:**a java.lang.Object holding the OUT parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.2 **See Also:**[setObject(java.lang.String, java.lang.Object, int, int)](http://docs.google.com/java/sql/CallableStatement.html#setObject(java.lang.String,%20java.lang.Object,%20int,%20int))

### getRef

[Ref](http://docs.google.com/java/sql/Ref.html) **getRef**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC REF(<structured-type>) parameter as a [Ref](http://docs.google.com/java/sql/Ref.html) object in the Java programming language.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value as a Ref object in the Java programming language. If the value was SQL NULL, the value null is returned. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.2

### getBlob

[Blob](http://docs.google.com/java/sql/Blob.html) **getBlob**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC BLOB parameter as a [Blob](http://docs.google.com/java/sql/Blob.html) object in the Java programming language.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value as a Blob object in the Java programming language. If the value was SQL NULL, the value null is returned. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.2

### getClob

[Clob](http://docs.google.com/java/sql/Clob.html) **getClob**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC CLOB parameter as a java.sql.Clob object in the Java programming language.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value as a Clob object in the Java programming language. If the value was SQL NULL, the value null is returned. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.2

### getArray

[Array](http://docs.google.com/java/sql/Array.html) **getArray**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC ARRAY parameter as an [Array](http://docs.google.com/java/sql/Array.html) object in the Java programming language.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value as an Array object in the Java programming language. If the value was SQL NULL, the value null is returned. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.2

### getDate

[Date](http://docs.google.com/java/sql/Date.html) **getDate**(int parameterIndex,  
 [Calendar](http://docs.google.com/java/util/Calendar.html) cal)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC DATE parameter as a java.sql.Date object, using the given Calendar object to construct the date. With a Calendar object, the driver can calculate the date taking into account a custom timezone and locale. If no Calendar object is specified, the driver uses the default timezone and locale.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so oncal - the Calendar object the driver will use to construct the date **Returns:**the parameter value. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**Since:** 1.2 **See Also:**[setDate(java.lang.String, java.sql.Date)](http://docs.google.com/java/sql/CallableStatement.html#setDate(java.lang.String,%20java.sql.Date))

### getTime

[Time](http://docs.google.com/java/sql/Time.html) **getTime**(int parameterIndex,  
 [Calendar](http://docs.google.com/java/util/Calendar.html) cal)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC TIME parameter as a java.sql.Time object, using the given Calendar object to construct the time. With a Calendar object, the driver can calculate the time taking into account a custom timezone and locale. If no Calendar object is specified, the driver uses the default timezone and locale.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so oncal - the Calendar object the driver will use to construct the time **Returns:**the parameter value; if the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**Since:** 1.2 **See Also:**[setTime(java.lang.String, java.sql.Time)](http://docs.google.com/java/sql/CallableStatement.html#setTime(java.lang.String,%20java.sql.Time))

### getTimestamp

[Timestamp](http://docs.google.com/java/sql/Timestamp.html) **getTimestamp**(int parameterIndex,  
 [Calendar](http://docs.google.com/java/util/Calendar.html) cal)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC TIMESTAMP parameter as a java.sql.Timestamp object, using the given Calendar object to construct the Timestamp object. With a Calendar object, the driver can calculate the timestamp taking into account a custom timezone and locale. If no Calendar object is specified, the driver uses the default timezone and locale.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so oncal - the Calendar object the driver will use to construct the timestamp **Returns:**the parameter value. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**Since:** 1.2 **See Also:**[setTimestamp(java.lang.String, java.sql.Timestamp)](http://docs.google.com/java/sql/CallableStatement.html#setTimestamp(java.lang.String,%20java.sql.Timestamp))

### registerOutParameter

void **registerOutParameter**(int parameterIndex,  
 int sqlType,  
 [String](http://docs.google.com/java/lang/String.html) typeName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Registers the designated output parameter. This version of the method registerOutParameter should be used for a user-defined or REF output parameter. Examples of user-defined types include: STRUCT, DISTINCT, JAVA\_OBJECT, and named array types.

All OUT parameters must be registered before a stored procedure is executed.

For a user-defined parameter, the fully-qualified SQL type name of the parameter should also be given, while a REF parameter requires that the fully-qualified type name of the referenced type be given. A JDBC driver that does not need the type code and type name information may ignore it. To be portable, however, applications should always provide these values for user-defined and REF parameters. Although it is intended for user-defined and REF parameters, this method may be used to register a parameter of any JDBC type. If the parameter does not have a user-defined or REF type, the *typeName* parameter is ignored.

**Note:** When reading the value of an out parameter, you must use the getter method whose Java type corresponds to the parameter's registered SQL type.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2,...sqlType - a value from [Types](http://docs.google.com/java/sql/Types.html)typeName - the fully-qualified name of an SQL structured type **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if sqlType is a ARRAY, BLOB, CLOB, DATALINK, JAVA\_OBJECT, NCHAR, NCLOB, NVARCHAR, LONGNVARCHAR, REF, ROWID, SQLXML or STRUCT data type and the JDBC driver does not support this data type**Since:** 1.2 **See Also:**[Types](http://docs.google.com/java/sql/Types.html)

### registerOutParameter

void **registerOutParameter**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 int sqlType)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Registers the OUT parameter named parameterName to the JDBC type sqlType. All OUT parameters must be registered before a stored procedure is executed.

The JDBC type specified by sqlType for an OUT parameter determines the Java type that must be used in the get method to read the value of that parameter.

If the JDBC type expected to be returned to this output parameter is specific to this particular database, sqlType should be java.sql.Types.OTHER. The method [getObject(int)](http://docs.google.com/java/sql/CallableStatement.html#getObject(int)) retrieves the value.

**Parameters:**parameterName - the name of the parametersqlType - the JDBC type code defined by java.sql.Types. If the parameter is of JDBC type NUMERIC or DECIMAL, the version of registerOutParameter that accepts a scale value should be used. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if sqlType is a ARRAY, BLOB, CLOB, DATALINK, JAVA\_OBJECT, NCHAR, NCLOB, NVARCHAR, LONGNVARCHAR, REF, ROWID, SQLXML or STRUCT data type and the JDBC driver does not support this data type or if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[Types](http://docs.google.com/java/sql/Types.html)

### registerOutParameter

void **registerOutParameter**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 int sqlType,  
 int scale)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Registers the parameter named parameterName to be of JDBC type sqlType. All OUT parameters must be registered before a stored procedure is executed.

The JDBC type specified by sqlType for an OUT parameter determines the Java type that must be used in the get method to read the value of that parameter.

This version of registerOutParameter should be used when the parameter is of JDBC type NUMERIC or DECIMAL.

**Parameters:**parameterName - the name of the parametersqlType - SQL type code defined by java.sql.Types.scale - the desired number of digits to the right of the decimal point. It must be greater than or equal to zero. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if sqlType is a ARRAY, BLOB, CLOB, DATALINK, JAVA\_OBJECT, NCHAR, NCLOB, NVARCHAR, LONGNVARCHAR, REF, ROWID, SQLXML or STRUCT data type and the JDBC driver does not support this data type or if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[Types](http://docs.google.com/java/sql/Types.html)

### registerOutParameter

void **registerOutParameter**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 int sqlType,  
 [String](http://docs.google.com/java/lang/String.html) typeName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Registers the designated output parameter. This version of the method registerOutParameter should be used for a user-named or REF output parameter. Examples of user-named types include: STRUCT, DISTINCT, JAVA\_OBJECT, and named array types.

All OUT parameters must be registered before a stored procedure is executed.

For a user-named parameter the fully-qualified SQL type name of the parameter should also be given, while a REF parameter requires that the fully-qualified type name of the referenced type be given. A JDBC driver that does not need the type code and type name information may ignore it. To be portable, however, applications should always provide these values for user-named and REF parameters. Although it is intended for user-named and REF parameters, this method may be used to register a parameter of any JDBC type. If the parameter does not have a user-named or REF type, the typeName parameter is ignored.

**Note:** When reading the value of an out parameter, you must use the getXXX method whose Java type XXX corresponds to the parameter's registered SQL type.

**Parameters:**parameterName - the name of the parametersqlType - a value from [Types](http://docs.google.com/java/sql/Types.html)typeName - the fully-qualified name of an SQL structured type **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if sqlType is a ARRAY, BLOB, CLOB, DATALINK, JAVA\_OBJECT, NCHAR, NCLOB, NVARCHAR, LONGNVARCHAR, REF, ROWID, SQLXML or STRUCT data type and the JDBC driver does not support this data type or if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[Types](http://docs.google.com/java/sql/Types.html)

### getURL

[URL](http://docs.google.com/java/net/URL.html) **getURL**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC DATALINK parameter as a java.net.URL object.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2,... **Returns:**a java.net.URL object that represents the JDBC DATALINK value used as the designated parameter **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs, this method is called on a closed CallableStatement, or if the URL being returned is not a valid URL on the Java platform [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setURL(java.lang.String, java.net.URL)](http://docs.google.com/java/sql/CallableStatement.html#setURL(java.lang.String,%20java.net.URL))

### setURL

void **setURL**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [URL](http://docs.google.com/java/net/URL.html) val)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given java.net.URL object. The driver converts this to an SQL DATALINK value when it sends it to the database.

**Parameters:**parameterName - the name of the parameterval - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs; this method is called on a closed CallableStatement or if a URL is malformed [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getURL(int)](http://docs.google.com/java/sql/CallableStatement.html#getURL(int))

### setNull

void **setNull**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 int sqlType)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to SQL NULL.

**Note:** You must specify the parameter's SQL type.

**Parameters:**parameterName - the name of the parametersqlType - the SQL type code defined in java.sql.Types **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4

### setBoolean

void **setBoolean**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 boolean x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given Java boolean value. The driver converts this to an SQL BIT or BOOLEAN value when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getBoolean(int)](http://docs.google.com/java/sql/CallableStatement.html#getBoolean(int))

### setByte

void **setByte**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 byte x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given Java byte value. The driver converts this to an SQL TINYINT value when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getByte(int)](http://docs.google.com/java/sql/CallableStatement.html#getByte(int))

### setShort

void **setShort**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 short x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given Java short value. The driver converts this to an SQL SMALLINT value when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getShort(int)](http://docs.google.com/java/sql/CallableStatement.html#getShort(int))

### setInt

void **setInt**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 int x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given Java int value. The driver converts this to an SQL INTEGER value when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getInt(int)](http://docs.google.com/java/sql/CallableStatement.html#getInt(int))

### setLong

void **setLong**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 long x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given Java long value. The driver converts this to an SQL BIGINT value when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getLong(int)](http://docs.google.com/java/sql/CallableStatement.html#getLong(int))

### setFloat

void **setFloat**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 float x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given Java float value. The driver converts this to an SQL FLOAT value when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getFloat(int)](http://docs.google.com/java/sql/CallableStatement.html#getFloat(int))

### setDouble

void **setDouble**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 double x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given Java double value. The driver converts this to an SQL DOUBLE value when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getDouble(int)](http://docs.google.com/java/sql/CallableStatement.html#getDouble(int))

### setBigDecimal

void **setBigDecimal**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [BigDecimal](http://docs.google.com/java/math/BigDecimal.html) x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given java.math.BigDecimal value. The driver converts this to an SQL NUMERIC value when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getBigDecimal(int, int)](http://docs.google.com/java/sql/CallableStatement.html#getBigDecimal(int,%20int))

### setString

void **setString**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [String](http://docs.google.com/java/lang/String.html) x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given Java String value. The driver converts this to an SQL VARCHAR or LONGVARCHAR value (depending on the argument's size relative to the driver's limits on VARCHAR values) when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getString(int)](http://docs.google.com/java/sql/CallableStatement.html#getString(int))

### setBytes

void **setBytes**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 byte[] x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given Java array of bytes. The driver converts this to an SQL VARBINARY or LONGVARBINARY (depending on the argument's size relative to the driver's limits on VARBINARY values) when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getBytes(int)](http://docs.google.com/java/sql/CallableStatement.html#getBytes(int))

### setDate

void **setDate**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Date](http://docs.google.com/java/sql/Date.html) x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given java.sql.Date value using the default time zone of the virtual machine that is running the application. The driver converts this to an SQL DATE value when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getDate(int)](http://docs.google.com/java/sql/CallableStatement.html#getDate(int))

### setTime

void **setTime**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Time](http://docs.google.com/java/sql/Time.html) x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given java.sql.Time value. The driver converts this to an SQL TIME value when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getTime(int)](http://docs.google.com/java/sql/CallableStatement.html#getTime(int))

### setTimestamp

void **setTimestamp**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Timestamp](http://docs.google.com/java/sql/Timestamp.html) x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given java.sql.Timestamp value. The driver converts this to an SQL TIMESTAMP value when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getTimestamp(int)](http://docs.google.com/java/sql/CallableStatement.html#getTimestamp(int))

### setAsciiStream

void **setAsciiStream**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [InputStream](http://docs.google.com/java/io/InputStream.html) x,  
 int length)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given input stream, which will have the specified number of bytes. When a very large ASCII value is input to a LONGVARCHAR parameter, it may be more practical to send it via a java.io.InputStream. Data will be read from the stream as needed until end-of-file is reached. The JDBC driver will do any necessary conversion from ASCII to the database char format.

**Note:** This stream object can either be a standard Java stream object or your own subclass that implements the standard interface.

**Parameters:**parameterName - the name of the parameterx - the Java input stream that contains the ASCII parameter valuelength - the number of bytes in the stream **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4

### setBinaryStream

void **setBinaryStream**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [InputStream](http://docs.google.com/java/io/InputStream.html) x,  
 int length)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given input stream, which will have the specified number of bytes. When a very large binary value is input to a LONGVARBINARY parameter, it may be more practical to send it via a java.io.InputStream object. The data will be read from the stream as needed until end-of-file is reached.

**Note:** This stream object can either be a standard Java stream object or your own subclass that implements the standard interface.

**Parameters:**parameterName - the name of the parameterx - the java input stream which contains the binary parameter valuelength - the number of bytes in the stream **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4

### setObject

void **setObject**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Object](http://docs.google.com/java/lang/Object.html) x,  
 int targetSqlType,  
 int scale)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the value of the designated parameter with the given object. The second argument must be an object type; for integral values, the java.lang equivalent objects should be used.

The given Java object will be converted to the given targetSqlType before being sent to the database. If the object has a custom mapping (is of a class implementing the interface SQLData), the JDBC driver should call the method SQLData.writeSQL to write it to the SQL data stream. If, on the other hand, the object is of a class implementing Ref, Blob, Clob, NClob, Struct, java.net.URL, or Array, the driver should pass it to the database as a value of the corresponding SQL type.

Note that this method may be used to pass datatabase- specific abstract data types.

**Parameters:**parameterName - the name of the parameterx - the object containing the input parameter valuetargetSqlType - the SQL type (as defined in java.sql.Types) to be sent to the database. The scale argument may further qualify this type.scale - for java.sql.Types.DECIMAL or java.sql.Types.NUMERIC types, this is the number of digits after the decimal point. For all other types, this value will be ignored. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if targetSqlType is a ARRAY, BLOB, CLOB, DATALINK, JAVA\_OBJECT, NCHAR, NCLOB, NVARCHAR, LONGNVARCHAR, REF, ROWID, SQLXML or STRUCT data type and the JDBC driver does not support this data type**Since:** 1.4 **See Also:**[Types](http://docs.google.com/java/sql/Types.html), [getObject(int)](http://docs.google.com/java/sql/CallableStatement.html#getObject(int))

### setObject

void **setObject**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Object](http://docs.google.com/java/lang/Object.html) x,  
 int targetSqlType)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the value of the designated parameter with the given object. This method is like the method setObject above, except that it assumes a scale of zero.

**Parameters:**parameterName - the name of the parameterx - the object containing the input parameter valuetargetSqlType - the SQL type (as defined in java.sql.Types) to be sent to the database **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if targetSqlType is a ARRAY, BLOB, CLOB, DATALINK, JAVA\_OBJECT, NCHAR, NCLOB, NVARCHAR, LONGNVARCHAR, REF, ROWID, SQLXML or STRUCT data type and the JDBC driver does not support this data type**Since:** 1.4 **See Also:**[getObject(int)](http://docs.google.com/java/sql/CallableStatement.html#getObject(int))

### setObject

void **setObject**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Object](http://docs.google.com/java/lang/Object.html) x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the value of the designated parameter with the given object. The second parameter must be of type Object; therefore, the java.lang equivalent objects should be used for built-in types.

The JDBC specification specifies a standard mapping from Java Object types to SQL types. The given argument will be converted to the corresponding SQL type before being sent to the database.

Note that this method may be used to pass datatabase- specific abstract data types, by using a driver-specific Java type. If the object is of a class implementing the interface SQLData, the JDBC driver should call the method SQLData.writeSQL to write it to the SQL data stream. If, on the other hand, the object is of a class implementing Ref, Blob, Clob, NClob, Struct, java.net.URL, or Array, the driver should pass it to the database as a value of the corresponding SQL type.

This method throws an exception if there is an ambiguity, for example, if the object is of a class implementing more than one of the interfaces named above.

**Note:** Not all databases allow for a non-typed Null to be sent to the backend. For maximum portability, the setNull or the setObject(String parameterName, Object x, int sqlType) method should be used instead of setObject(String parameterName, Object x).

**Parameters:**parameterName - the name of the parameterx - the object containing the input parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs, this method is called on a closed CallableStatement or if the given Object parameter is ambiguous [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getObject(int)](http://docs.google.com/java/sql/CallableStatement.html#getObject(int))

### setCharacterStream

void **setCharacterStream**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Reader](http://docs.google.com/java/io/Reader.html) reader,  
 int length)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given Reader object, which is the given number of characters long. When a very large UNICODE value is input to a LONGVARCHAR parameter, it may be more practical to send it via a java.io.Reader object. The data will be read from the stream as needed until end-of-file is reached. The JDBC driver will do any necessary conversion from UNICODE to the database char format.

**Note:** This stream object can either be a standard Java stream object or your own subclass that implements the standard interface.

**Parameters:**parameterName - the name of the parameterreader - the java.io.Reader object that contains the UNICODE data used as the designated parameterlength - the number of characters in the stream **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4

### setDate

void **setDate**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Date](http://docs.google.com/java/sql/Date.html) x,  
 [Calendar](http://docs.google.com/java/util/Calendar.html) cal)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given java.sql.Date value, using the given Calendar object. The driver uses the Calendar object to construct an SQL DATE value, which the driver then sends to the database. With a a Calendar object, the driver can calculate the date taking into account a custom timezone. If no Calendar object is specified, the driver uses the default timezone, which is that of the virtual machine running the application.

**Parameters:**parameterName - the name of the parameterx - the parameter valuecal - the Calendar object the driver will use to construct the date **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getDate(int)](http://docs.google.com/java/sql/CallableStatement.html#getDate(int))

### setTime

void **setTime**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Time](http://docs.google.com/java/sql/Time.html) x,  
 [Calendar](http://docs.google.com/java/util/Calendar.html) cal)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given java.sql.Time value, using the given Calendar object. The driver uses the Calendar object to construct an SQL TIME value, which the driver then sends to the database. With a a Calendar object, the driver can calculate the time taking into account a custom timezone. If no Calendar object is specified, the driver uses the default timezone, which is that of the virtual machine running the application.

**Parameters:**parameterName - the name of the parameterx - the parameter valuecal - the Calendar object the driver will use to construct the time **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getTime(int)](http://docs.google.com/java/sql/CallableStatement.html#getTime(int))

### setTimestamp

void **setTimestamp**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Timestamp](http://docs.google.com/java/sql/Timestamp.html) x,  
 [Calendar](http://docs.google.com/java/util/Calendar.html) cal)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given java.sql.Timestamp value, using the given Calendar object. The driver uses the Calendar object to construct an SQL TIMESTAMP value, which the driver then sends to the database. With a a Calendar object, the driver can calculate the timestamp taking into account a custom timezone. If no Calendar object is specified, the driver uses the default timezone, which is that of the virtual machine running the application.

**Parameters:**parameterName - the name of the parameterx - the parameter valuecal - the Calendar object the driver will use to construct the timestamp **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[getTimestamp(int)](http://docs.google.com/java/sql/CallableStatement.html#getTimestamp(int))

### setNull

void **setNull**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 int sqlType,  
 [String](http://docs.google.com/java/lang/String.html) typeName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to SQL NULL. This version of the method setNull should be used for user-defined types and REF type parameters. Examples of user-defined types include: STRUCT, DISTINCT, JAVA\_OBJECT, and named array types.

**Note:** To be portable, applications must give the SQL type code and the fully-qualified SQL type name when specifying a NULL user-defined or REF parameter. In the case of a user-defined type the name is the type name of the parameter itself. For a REF parameter, the name is the type name of the referenced type.

Although it is intended for user-defined and Ref parameters, this method may be used to set a null parameter of any JDBC type. If the parameter does not have a user-defined or REF type, the given typeName is ignored.

**Parameters:**parameterName - the name of the parametersqlType - a value from java.sql.TypestypeName - the fully-qualified name of an SQL user-defined type; ignored if the parameter is not a user-defined type or SQL REF value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4

### getString

[String](http://docs.google.com/java/lang/String.html) **getString**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC CHAR, VARCHAR, or LONGVARCHAR parameter as a String in the Java programming language.

For the fixed-length type JDBC CHAR, the String object returned has exactly the same value the SQL CHAR value had in the database, including any padding added by the database.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setString(java.lang.String, java.lang.String)](http://docs.google.com/java/sql/CallableStatement.html#setString(java.lang.String,%20java.lang.String))

### getBoolean

boolean **getBoolean**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC BIT or BOOLEAN parameter as a boolean in the Java programming language.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value. If the value is SQL NULL, the result is false. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setBoolean(java.lang.String, boolean)](http://docs.google.com/java/sql/CallableStatement.html#setBoolean(java.lang.String,%20boolean))

### getByte

byte **getByte**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC TINYINT parameter as a byte in the Java programming language.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value. If the value is SQL NULL, the result is 0. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setByte(java.lang.String, byte)](http://docs.google.com/java/sql/CallableStatement.html#setByte(java.lang.String,%20byte))

### getShort

short **getShort**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC SMALLINT parameter as a short in the Java programming language.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value. If the value is SQL NULL, the result is 0. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setShort(java.lang.String, short)](http://docs.google.com/java/sql/CallableStatement.html#setShort(java.lang.String,%20short))

### getInt

int **getInt**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC INTEGER parameter as an int in the Java programming language.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value. If the value is SQL NULL, the result is 0. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setInt(java.lang.String, int)](http://docs.google.com/java/sql/CallableStatement.html#setInt(java.lang.String,%20int))

### getLong

long **getLong**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC BIGINT parameter as a long in the Java programming language.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value. If the value is SQL NULL, the result is 0. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setLong(java.lang.String, long)](http://docs.google.com/java/sql/CallableStatement.html#setLong(java.lang.String,%20long))

### getFloat

float **getFloat**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC FLOAT parameter as a float in the Java programming language.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value. If the value is SQL NULL, the result is 0. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setFloat(java.lang.String, float)](http://docs.google.com/java/sql/CallableStatement.html#setFloat(java.lang.String,%20float))

### getDouble

double **getDouble**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC DOUBLE parameter as a double in the Java programming language.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value. If the value is SQL NULL, the result is 0. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setDouble(java.lang.String, double)](http://docs.google.com/java/sql/CallableStatement.html#setDouble(java.lang.String,%20double))

### getBytes

byte[] **getBytes**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC BINARY or VARBINARY parameter as an array of byte values in the Java programming language.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setBytes(java.lang.String, byte[])](http://docs.google.com/java/sql/CallableStatement.html#setBytes(java.lang.String,%20byte%5B%5D))

### getDate

[Date](http://docs.google.com/java/sql/Date.html) **getDate**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC DATE parameter as a java.sql.Date object.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setDate(java.lang.String, java.sql.Date)](http://docs.google.com/java/sql/CallableStatement.html#setDate(java.lang.String,%20java.sql.Date))

### getTime

[Time](http://docs.google.com/java/sql/Time.html) **getTime**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC TIME parameter as a java.sql.Time object.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setTime(java.lang.String, java.sql.Time)](http://docs.google.com/java/sql/CallableStatement.html#setTime(java.lang.String,%20java.sql.Time))

### getTimestamp

[Timestamp](http://docs.google.com/java/sql/Timestamp.html) **getTimestamp**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC TIMESTAMP parameter as a java.sql.Timestamp object.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setTimestamp(java.lang.String, java.sql.Timestamp)](http://docs.google.com/java/sql/CallableStatement.html#setTimestamp(java.lang.String,%20java.sql.Timestamp))

### getObject

[Object](http://docs.google.com/java/lang/Object.html) **getObject**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a parameter as an Object in the Java programming language. If the value is an SQL NULL, the driver returns a Java null.

This method returns a Java object whose type corresponds to the JDBC type that was registered for this parameter using the method registerOutParameter. By registering the target JDBC type as java.sql.Types.OTHER, this method can be used to read database-specific abstract data types.

**Parameters:**parameterName - the name of the parameter **Returns:**A java.lang.Object holding the OUT parameter value. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[Types](http://docs.google.com/java/sql/Types.html), [setObject(java.lang.String, java.lang.Object, int, int)](http://docs.google.com/java/sql/CallableStatement.html#setObject(java.lang.String,%20java.lang.Object,%20int,%20int))

### getBigDecimal

[BigDecimal](http://docs.google.com/java/math/BigDecimal.html) **getBigDecimal**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC NUMERIC parameter as a java.math.BigDecimal object with as many digits to the right of the decimal point as the value contains.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value in full precision. If the value is SQL NULL, the result is null. **Throws:** SQLExceptionif - parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method [SQLException](http://docs.google.com/java/sql/SQLException.html)**Since:** 1.4 **See Also:**[setBigDecimal(java.lang.String, java.math.BigDecimal)](http://docs.google.com/java/sql/CallableStatement.html#setBigDecimal(java.lang.String,%20java.math.BigDecimal))

### getObject

[Object](http://docs.google.com/java/lang/Object.html) **getObject**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),[Class](http://docs.google.com/java/lang/Class.html)<?>> map)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Returns an object representing the value of OUT parameter parameterName and uses map for the custom mapping of the parameter value.

This method returns a Java object whose type corresponds to the JDBC type that was registered for this parameter using the method registerOutParameter. By registering the target JDBC type as java.sql.Types.OTHER, this method can be used to read database-specific abstract data types.

**Parameters:**parameterName - the name of the parametermap - the mapping from SQL type names to Java classes **Returns:**a java.lang.Object holding the OUT parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setObject(java.lang.String, java.lang.Object, int, int)](http://docs.google.com/java/sql/CallableStatement.html#setObject(java.lang.String,%20java.lang.Object,%20int,%20int))

### getRef

[Ref](http://docs.google.com/java/sql/Ref.html) **getRef**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC REF(<structured-type>) parameter as a [Ref](http://docs.google.com/java/sql/Ref.html) object in the Java programming language.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value as a Ref object in the Java programming language. If the value was SQL NULL, the value null is returned. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4

### getBlob

[Blob](http://docs.google.com/java/sql/Blob.html) **getBlob**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC BLOB parameter as a [Blob](http://docs.google.com/java/sql/Blob.html) object in the Java programming language.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value as a Blob object in the Java programming language. If the value was SQL NULL, the value null is returned. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4

### getClob

[Clob](http://docs.google.com/java/sql/Clob.html) **getClob**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC CLOB parameter as a java.sql.Clob object in the Java programming language.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value as a Clob object in the Java programming language. If the value was SQL NULL, the value null is returned. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4

### getArray

[Array](http://docs.google.com/java/sql/Array.html) **getArray**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC ARRAY parameter as an [Array](http://docs.google.com/java/sql/Array.html) object in the Java programming language.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value as an Array object in Java programming language. If the value was SQL NULL, the value null is returned. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4

### getDate

[Date](http://docs.google.com/java/sql/Date.html) **getDate**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Calendar](http://docs.google.com/java/util/Calendar.html) cal)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC DATE parameter as a java.sql.Date object, using the given Calendar object to construct the date. With a Calendar object, the driver can calculate the date taking into account a custom timezone and locale. If no Calendar object is specified, the driver uses the default timezone and locale.

**Parameters:**parameterName - the name of the parametercal - the Calendar object the driver will use to construct the date **Returns:**the parameter value. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setDate(java.lang.String, java.sql.Date)](http://docs.google.com/java/sql/CallableStatement.html#setDate(java.lang.String,%20java.sql.Date))

### getTime

[Time](http://docs.google.com/java/sql/Time.html) **getTime**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Calendar](http://docs.google.com/java/util/Calendar.html) cal)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC TIME parameter as a java.sql.Time object, using the given Calendar object to construct the time. With a Calendar object, the driver can calculate the time taking into account a custom timezone and locale. If no Calendar object is specified, the driver uses the default timezone and locale.

**Parameters:**parameterName - the name of the parametercal - the Calendar object the driver will use to construct the time **Returns:**the parameter value; if the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setTime(java.lang.String, java.sql.Time)](http://docs.google.com/java/sql/CallableStatement.html#setTime(java.lang.String,%20java.sql.Time))

### getTimestamp

[Timestamp](http://docs.google.com/java/sql/Timestamp.html) **getTimestamp**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Calendar](http://docs.google.com/java/util/Calendar.html) cal)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC TIMESTAMP parameter as a java.sql.Timestamp object, using the given Calendar object to construct the Timestamp object. With a Calendar object, the driver can calculate the timestamp taking into account a custom timezone and locale. If no Calendar object is specified, the driver uses the default timezone and locale.

**Parameters:**parameterName - the name of the parametercal - the Calendar object the driver will use to construct the timestamp **Returns:**the parameter value. If the value is SQL NULL, the result is null. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setTimestamp(java.lang.String, java.sql.Timestamp)](http://docs.google.com/java/sql/CallableStatement.html#setTimestamp(java.lang.String,%20java.sql.Timestamp))

### getURL

[URL](http://docs.google.com/java/net/URL.html) **getURL**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC DATALINK parameter as a java.net.URL object.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value as a java.net.URL object in the Java programming language. If the value was SQL NULL, the value null is returned. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs, this method is called on a closed CallableStatement, or if there is a problem with the URL [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.4 **See Also:**[setURL(java.lang.String, java.net.URL)](http://docs.google.com/java/sql/CallableStatement.html#setURL(java.lang.String,%20java.net.URL))

### getRowId

[RowId](http://docs.google.com/java/sql/RowId.html) **getRowId**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC ROWID parameter as a java.sql.RowId object.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2,... **Returns:**a RowId object that represents the JDBC ROWID value is used as the designated parameter. If the parameter contains a SQL NULL, then a null value is returned. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### getRowId

[RowId](http://docs.google.com/java/sql/RowId.html) **getRowId**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC ROWID parameter as a java.sql.RowId object.

**Parameters:**parameterName - the name of the parameter **Returns:**a RowId object that represents the JDBC ROWID value is used as the designated parameter. If the parameter contains a SQL NULL, then a null value is returned. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setRowId

void **setRowId**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [RowId](http://docs.google.com/java/sql/RowId.html) x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given java.sql.RowId object. The driver converts this to a SQL ROWID when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setNString

void **setNString**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [String](http://docs.google.com/java/lang/String.html) value)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given String object. The driver converts this to a SQL NCHAR or NVARCHAR or LONGNVARCHAR

**Parameters:**parameterName - the name of the parameter to be setvalue - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if the driver does not support national character sets; if the driver can detect that a data conversion error could occur; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setNCharacterStream

void **setNCharacterStream**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Reader](http://docs.google.com/java/io/Reader.html) value,  
 long length)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to a Reader object. The Reader reads the data till end-of-file is reached. The driver does the necessary conversion from Java character format to the national character set in the database.

**Parameters:**parameterName - the name of the parameter to be setvalue - the parameter valuelength - the number of characters in the parameter data. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if the driver does not support national character sets; if the driver can detect that a data conversion error could occur; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setNClob

void **setNClob**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [NClob](http://docs.google.com/java/sql/NClob.html) value)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to a java.sql.NClob object. The object implements the java.sql.NClob interface. This NClob object maps to a SQL NCLOB.

**Parameters:**parameterName - the name of the parameter to be setvalue - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if the driver does not support national character sets; if the driver can detect that a data conversion error could occur; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setClob

void **setClob**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Reader](http://docs.google.com/java/io/Reader.html) reader,  
 long length)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to a Reader object. The reader must contain the number of characters specified by length otherwise a SQLException will be generated when the CallableStatement is executed. This method differs from the setCharacterStream (int, Reader, int) method because it informs the driver that the parameter value should be sent to the server as a CLOB. When the setCharacterStream method is used, the driver may have to do extra work to determine whether the parameter data should be send to the server as a LONGVARCHAR or a CLOB

**Parameters:**parameterName - the name of the parameter to be setreader - An object that contains the data to set the parameter value to.length - the number of characters in the parameter data. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if the length specified is less than zero; a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setBlob

void **setBlob**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [InputStream](http://docs.google.com/java/io/InputStream.html) inputStream,  
 long length)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to a InputStream object. The inputstream must contain the number of characters specified by length, otherwise a SQLException will be generated when the CallableStatement is executed. This method differs from the setBinaryStream (int, InputStream, int) method because it informs the driver that the parameter value should be sent to the server as a BLOB. When the setBinaryStream method is used, the driver may have to do extra work to determine whether the parameter data should be sent to the server as a LONGVARBINARY or a BLOB

**Parameters:**parameterName - the name of the parameter to be set the second is 2, ...inputStream - An object that contains the data to set the parameter value to.length - the number of bytes in the parameter data. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if the length specified is less than zero; if the number of bytes in the inputstream does not match the specfied length; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setNClob

void **setNClob**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Reader](http://docs.google.com/java/io/Reader.html) reader,  
 long length)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to a Reader object. The reader must contain the number of characters specified by length otherwise a SQLException will be generated when the CallableStatement is executed. This method differs from the setCharacterStream (int, Reader, int) method because it informs the driver that the parameter value should be sent to the server as a NCLOB. When the setCharacterStream method is used, the driver may have to do extra work to determine whether the parameter data should be send to the server as a LONGNVARCHAR or a NCLOB

**Parameters:**parameterName - the name of the parameter to be setreader - An object that contains the data to set the parameter value to.length - the number of characters in the parameter data. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if the length specified is less than zero; if the driver does not support national character sets; if the driver can detect that a data conversion error could occur; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### getNClob

[NClob](http://docs.google.com/java/sql/NClob.html) **getNClob**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated JDBC NCLOB parameter as a java.sql.NClob object in the Java programming language.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, and so on **Returns:**the parameter value as a NClob object in the Java programming language. If the value was SQL NULL, the value null is returned. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if the driver does not support national character sets; if the driver can detect that a data conversion error could occur; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### getNClob

[NClob](http://docs.google.com/java/sql/NClob.html) **getNClob**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of a JDBC NCLOB parameter as a java.sql.NClob object in the Java programming language.

**Parameters:**parameterName - the name of the parameter **Returns:**the parameter value as a NClob object in the Java programming language. If the value was SQL NULL, the value null is returned. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if the driver does not support national character sets; if the driver can detect that a data conversion error could occur; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setSQLXML

void **setSQLXML**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [SQLXML](http://docs.google.com/java/sql/SQLXML.html) xmlObject)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given java.sql.SQLXML object. The driver converts this to an SQL XML value when it sends it to the database.

**Parameters:**parameterName - the name of the parameterxmlObject - a SQLXML object that maps an SQL XML value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs; this method is called on a closed CallableStatement or the java.xml.transform.Result, Writer or OutputStream has not been closed for the SQLXML object [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### getSQLXML

[SQLXML](http://docs.google.com/java/sql/SQLXML.html) **getSQLXML**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated SQL XML parameter as a java.sql.SQLXML object in the Java programming language.

**Parameters:**parameterIndex - index of the first parameter is 1, the second is 2, ... **Returns:**a SQLXML object that maps an SQL XML value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### getSQLXML

[SQLXML](http://docs.google.com/java/sql/SQLXML.html) **getSQLXML**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated SQL XML parameter as a java.sql.SQLXML object in the Java programming language.

**Parameters:**parameterName - the name of the parameter **Returns:**a SQLXML object that maps an SQL XML value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### getNString

[String](http://docs.google.com/java/lang/String.html) **getNString**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated NCHAR, NVARCHAR or LONGNVARCHAR parameter as a String in the Java programming language.

For the fixed-length type JDBC NCHAR, the String object returned has exactly the same value the SQL NCHAR value had in the database, including any padding added by the database.

**Parameters:**parameterIndex - index of the first parameter is 1, the second is 2, ... **Returns:**a String object that maps an NCHAR, NVARCHAR or LONGNVARCHAR value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6 **See Also:**[setNString(java.lang.String, java.lang.String)](http://docs.google.com/java/sql/CallableStatement.html#setNString(java.lang.String,%20java.lang.String))

### getNString

[String](http://docs.google.com/java/lang/String.html) **getNString**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated NCHAR, NVARCHAR or LONGNVARCHAR parameter as a String in the Java programming language.

For the fixed-length type JDBC NCHAR, the String object returned has exactly the same value the SQL NCHAR value had in the database, including any padding added by the database.

**Parameters:**parameterName - the name of the parameter **Returns:**a String object that maps an NCHAR, NVARCHAR or LONGNVARCHAR value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6 **See Also:**[setNString(java.lang.String, java.lang.String)](http://docs.google.com/java/sql/CallableStatement.html#setNString(java.lang.String,%20java.lang.String))

### getNCharacterStream

[Reader](http://docs.google.com/java/io/Reader.html) **getNCharacterStream**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated parameter as a java.io.Reader object in the Java programming language. It is intended for use when accessing NCHAR,NVARCHAR and LONGNVARCHAR parameters.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, ... **Returns:**a java.io.Reader object that contains the parameter value; if the value is SQL NULL, the value returned is null in the Java programming language. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### getNCharacterStream

[Reader](http://docs.google.com/java/io/Reader.html) **getNCharacterStream**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated parameter as a java.io.Reader object in the Java programming language. It is intended for use when accessing NCHAR,NVARCHAR and LONGNVARCHAR parameters.

**Parameters:**parameterName - the name of the parameter **Returns:**a java.io.Reader object that contains the parameter value; if the value is SQL NULL, the value returned is null in the Java programming language **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### getCharacterStream

[Reader](http://docs.google.com/java/io/Reader.html) **getCharacterStream**(int parameterIndex)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated parameter as a java.io.Reader object in the Java programming language.

**Parameters:**parameterIndex - the first parameter is 1, the second is 2, ... **Returns:**a java.io.Reader object that contains the parameter value; if the value is SQL NULL, the value returned is null in the Java programming language. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if the parameterIndex is not valid; if a database access error occurs or this method is called on a closed CallableStatement**Since:** 1.6

### getCharacterStream

[Reader](http://docs.google.com/java/io/Reader.html) **getCharacterStream**([String](http://docs.google.com/java/lang/String.html) parameterName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the value of the designated parameter as a java.io.Reader object in the Java programming language.

**Parameters:**parameterName - the name of the parameter **Returns:**a java.io.Reader object that contains the parameter value; if the value is SQL NULL, the value returned is null in the Java programming language **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setBlob

void **setBlob**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Blob](http://docs.google.com/java/sql/Blob.html) x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given java.sql.Blob object. The driver converts this to an SQL BLOB value when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - a Blob object that maps an SQL BLOB value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setClob

void **setClob**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Clob](http://docs.google.com/java/sql/Clob.html) x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given java.sql.Clob object. The driver converts this to an SQL CLOB value when it sends it to the database.

**Parameters:**parameterName - the name of the parameterx - a Clob object that maps an SQL CLOB value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setAsciiStream

void **setAsciiStream**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [InputStream](http://docs.google.com/java/io/InputStream.html) x,  
 long length)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given input stream, which will have the specified number of bytes. When a very large ASCII value is input to a LONGVARCHAR parameter, it may be more practical to send it via a java.io.InputStream. Data will be read from the stream as needed until end-of-file is reached. The JDBC driver will do any necessary conversion from ASCII to the database char format.

**Note:** This stream object can either be a standard Java stream object or your own subclass that implements the standard interface.

**Parameters:**parameterName - the name of the parameterx - the Java input stream that contains the ASCII parameter valuelength - the number of bytes in the stream **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setBinaryStream

void **setBinaryStream**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [InputStream](http://docs.google.com/java/io/InputStream.html) x,  
 long length)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given input stream, which will have the specified number of bytes. When a very large binary value is input to a LONGVARBINARY parameter, it may be more practical to send it via a java.io.InputStream object. The data will be read from the stream as needed until end-of-file is reached.

**Note:** This stream object can either be a standard Java stream object or your own subclass that implements the standard interface.

**Parameters:**parameterName - the name of the parameterx - the java input stream which contains the binary parameter valuelength - the number of bytes in the stream **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setCharacterStream

void **setCharacterStream**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Reader](http://docs.google.com/java/io/Reader.html) reader,  
 long length)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given Reader object, which is the given number of characters long. When a very large UNICODE value is input to a LONGVARCHAR parameter, it may be more practical to send it via a java.io.Reader object. The data will be read from the stream as needed until end-of-file is reached. The JDBC driver will do any necessary conversion from UNICODE to the database char format.

**Note:** This stream object can either be a standard Java stream object or your own subclass that implements the standard interface.

**Parameters:**parameterName - the name of the parameterreader - the java.io.Reader object that contains the UNICODE data used as the designated parameterlength - the number of characters in the stream **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setAsciiStream

void **setAsciiStream**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [InputStream](http://docs.google.com/java/io/InputStream.html) x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given input stream. When a very large ASCII value is input to a LONGVARCHAR parameter, it may be more practical to send it via a java.io.InputStream. Data will be read from the stream as needed until end-of-file is reached. The JDBC driver will do any necessary conversion from ASCII to the database char format.

**Note:** This stream object can either be a standard Java stream object or your own subclass that implements the standard interface.

**Note:** Consult your JDBC driver documentation to determine if it might be more efficient to use a version of setAsciiStream which takes a length parameter.

**Parameters:**parameterName - the name of the parameterx - the Java input stream that contains the ASCII parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setBinaryStream

void **setBinaryStream**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [InputStream](http://docs.google.com/java/io/InputStream.html) x)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given input stream. When a very large binary value is input to a LONGVARBINARY parameter, it may be more practical to send it via a java.io.InputStream object. The data will be read from the stream as needed until end-of-file is reached.

**Note:** This stream object can either be a standard Java stream object or your own subclass that implements the standard interface.

**Note:** Consult your JDBC driver documentation to determine if it might be more efficient to use a version of setBinaryStream which takes a length parameter.

**Parameters:**parameterName - the name of the parameterx - the java input stream which contains the binary parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setCharacterStream

void **setCharacterStream**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Reader](http://docs.google.com/java/io/Reader.html) reader)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to the given Reader object. When a very large UNICODE value is input to a LONGVARCHAR parameter, it may be more practical to send it via a java.io.Reader object. The data will be read from the stream as needed until end-of-file is reached. The JDBC driver will do any necessary conversion from UNICODE to the database char format.

**Note:** This stream object can either be a standard Java stream object or your own subclass that implements the standard interface.

**Note:** Consult your JDBC driver documentation to determine if it might be more efficient to use a version of setCharacterStream which takes a length parameter.

**Parameters:**parameterName - the name of the parameterreader - the java.io.Reader object that contains the Unicode data **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setNCharacterStream

void **setNCharacterStream**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Reader](http://docs.google.com/java/io/Reader.html) value)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to a Reader object. The Reader reads the data till end-of-file is reached. The driver does the necessary conversion from Java character format to the national character set in the database.

**Note:** This stream object can either be a standard Java stream object or your own subclass that implements the standard interface.

**Note:** Consult your JDBC driver documentation to determine if it might be more efficient to use a version of setNCharacterStream which takes a length parameter.

**Parameters:**parameterName - the name of the parametervalue - the parameter value **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if the driver does not support national character sets; if the driver can detect that a data conversion error could occur; if a database access error occurs; or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setClob

void **setClob**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Reader](http://docs.google.com/java/io/Reader.html) reader)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to a Reader object. This method differs from the setCharacterStream (int, Reader) method because it informs the driver that the parameter value should be sent to the server as a CLOB. When the setCharacterStream method is used, the driver may have to do extra work to determine whether the parameter data should be send to the server as a LONGVARCHAR or a CLOB

**Note:** Consult your JDBC driver documentation to determine if it might be more efficient to use a version of setClob which takes a length parameter.

**Parameters:**parameterName - the name of the parameterreader - An object that contains the data to set the parameter value to. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setBlob

void **setBlob**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [InputStream](http://docs.google.com/java/io/InputStream.html) inputStream)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to a InputStream object. This method differs from the setBinaryStream (int, InputStream) method because it informs the driver that the parameter value should be sent to the server as a BLOB. When the setBinaryStream method is used, the driver may have to do extra work to determine whether the parameter data should be send to the server as a LONGVARBINARY or a BLOB

**Note:** Consult your JDBC driver documentation to determine if it might be more efficient to use a version of setBlob which takes a length parameter.

**Parameters:**parameterName - the name of the parameterinputStream - An object that contains the data to set the parameter value to. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

### setNClob

void **setNClob**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [Reader](http://docs.google.com/java/io/Reader.html) reader)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Sets the designated parameter to a Reader object. This method differs from the setCharacterStream (int, Reader) method because it informs the driver that the parameter value should be sent to the server as a NCLOB. When the setCharacterStream method is used, the driver may have to do extra work to determine whether the parameter data should be send to the server as a LONGNVARCHAR or a NCLOB

**Note:** Consult your JDBC driver documentation to determine if it might be more efficient to use a version of setNClob which takes a length parameter.

**Parameters:**parameterName - the name of the parameterreader - An object that contains the data to set the parameter value to. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if parameterName does not correspond to a named parameter; if the driver does not support national character sets; if the driver can detect that a data conversion error could occur; if a database access error occurs or this method is called on a closed CallableStatement [SQLFeatureNotSupportedException](http://docs.google.com/java/sql/SQLFeatureNotSupportedException.html) - if the JDBC driver does not support this method**Since:** 1.6

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/CallableStatement.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/sql/Blob.html)   [**NEXT CLASS**](http://docs.google.com/java/sql/ClientInfoStatus.html) | [**FRAMES**](http://docs.google.com/index.html?java/sql/CallableStatement.html)    [**NO FRAMES**](http://docs.google.com/CallableStatement.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#tyjcwt) | DETAIL: FIELD | CONSTR | [METHOD](#2s8eyo1) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).