Overview

olap4j is an open Java API for OLAP.

Resources:

- olap4j project at SourceForge
- Download specification
- Home page

Package
org.olap4j

Provides the core classes and interfaces of the olap4j API for accessing and processing OLAP data.

org.olap4j
Interface Axis

All Known Implementing Classes:

Standard

public interface **Axis**
extends

Enumeration of axis types.

The most commonly used values are `COLUMNS` (the first axis of a 2-dimensional query), `ROWS` (the second axis of a 2-dimensional query) and `FILTER` (also known as the slicer axis, denoted by a `WHERE` clause in an MDX statement).

Nested Class Summary

`class`	`Axis.Factory` Axis.Factory
`class`	`Axis.Standard` Axis.Standard

Field Summary

`public static final`	`CHAPTERS` Abbreviation for `Axis.Standard.FILTER`.
`public static final`	`COLUMNS` Abbreviation for `Axis.Standard.COLUMNS`.
`public static final`	`FILTER` Abbreviation for `Axis.Standard.FILTER`.
`public static final`	`NONE` **Deprecated.** *Will be removed before olap4j 1.0.*
`public static final`	`PAGES` Abbreviation for `Axis.Standard.PAGES`.
`public static final`	`ROWS` Abbreviation for `Axis.Standard.ROWS`.
`public static final`	`SECTIONS` Abbreviation for `Axis.Standard.CHAPTERS`.
`public static final`	`UNUSED` **Deprecated.** *Will be removed before olap4j 1.0.*

Method Summary

`int`	`axisOrdinal()` Returns the ordinal which is to be used for retrieving this axis from the `CellSet.getAxes()`, or retrieving its coordinate from `getCoordinateList()`.

`java.lang.String`	`getCaption(java.util.Locale locale)` Returns localized name for this Axis.
`boolean`	`isFilter()` Returns whether this is the filter (slicer) axis.
`java.lang.String`	`name()` Returns the name of this axis, e.g.

Fields

UNUSED

`public static final org.olap4j.Axis.Standard` **`UNUSED`**

Deprecated. *Will be removed before olap4j 1.0.*

NONE

`public static final org.olap4j.Axis.Standard` **`NONE`**

Deprecated. *Will be removed before olap4j 1.0.*

FILTER

`public static final org.olap4j.Axis.Standard` **`FILTER`**

Abbreviation for `Axis.Standard.FILTER`.

COLUMNS

`public static final org.olap4j.Axis.Standard` **`COLUMNS`**

Abbreviation for `Axis.Standard.COLUMNS`.

ROWS

`public static final org.olap4j.Axis.Standard` **`ROWS`**

Abbreviation for `Axis.Standard.ROWS`.

PAGES

`public static final org.olap4j.Axis.Standard` **`PAGES`**

Abbreviation for `Axis.Standard.PAGES`.

SECTIONS

`public static final org.olap4j.Axis.Standard` **`SECTIONS`**

Abbreviation for `Axis.Standard.CHAPTERS`.

CHAPTERS

`public static final org.olap4j.Axis.Standard **CHAPTERS**`

Abbreviation for `Axis.Standard.FILTER`.

Methods

name

`public java.lang.String **name**()`

Returns the name of this axis, e.g. "COLUMNS", "FILTER", "AXIS(17)".

Returns:
Name of the axis

isFilter

`public boolean **isFilter**()`

Returns whether this is the filter (slicer) axis.

Returns:
whether this is the filter axis

axisOrdinal

`public int **axisOrdinal**()`

Returns the ordinal which is to be used for retrieving this axis from the `CellSet.getAxes()`, or retrieving its coordinate from `getCoordinateList()`.

For example:

- -1 `FILTER`
- 0 `COLUMNS`
- 1 `ROWS`
- 2 `PAGES`
- 3 `CHAPTERS`
- 4 `SECTIONS`
- 5 `SECTIONS`
- 6 AXES(6)
- 123 AXES(123)

Returns:
ordinal of this axis

getCaption

`public java.lang.String **getCaption**(java.util.Locale locale)`

Returns localized name for this Axis.

Examples: "FILTER", "ROWS", "COLUMNS", "AXIS(10)".

Parameters:

(continued from last page)

`locale` - Locale for which to give the name

Returns:

localized name for this Axis

org.olap4j
Class Axis.Standard

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.Axis.Standard
```

All Implemented Interfaces:
Axis, java.io.Serializable, java.lang.Comparable

public static final class **Axis.Standard**
extends java.lang.Enum
implements java.lang.Comparable, java.io.Serializable, Axis

Enumeration of standard, named axes descriptors.

Field Summary

public static final	CHAPTERS CHAPTERS axis, also known as AXIS(3).
public static final	COLUMNS COLUMNS axis, also known as X axis and AXIS(0).
public static final	FILTER Filter axis, also known as the slicer axis.
public static final	PAGES PAGES axis, also known as AXIS(2).
public static final	ROWS ROWS axis, also known as Y axis and AXIS(1).
public static final	SECTIONS SECTIONS axis, also known as AXIS(4).

Fields inherited from interface org.olap4j.Axis

CHAPTERS, COLUMNS, FILTER, NONE, PAGES, ROWS, SECTIONS, UNUSED

Method Summary

int	axisOrdinal()
java.lang.String	getCaption(java.util.Locale locale)
boolean	isFilter()
static Axis.Standard	valueOf(java.lang.String name)
static Axis.Standard[]	values()

Methods inherited from class `java.lang.Enum`
`compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `java.lang.Comparable`
`compareTo`

Methods inherited from interface `org.olap4j.Axis`
`axisOrdinal, getCaption, isFilter, name`

Fields

FILTER

`public static final org.olap4j.Axis.Standard` **`FILTER`**

Filter axis, also known as the slicer axis.

COLUMNS

`public static final org.olap4j.Axis.Standard` **`COLUMNS`**

COLUMNS axis, also known as X axis and AXIS(0).

ROWS

`public static final org.olap4j.Axis.Standard` **`ROWS`**

ROWS axis, also known as Y axis and AXIS(1).

PAGES

`public static final org.olap4j.Axis.Standard` **`PAGES`**

PAGES axis, also known as AXIS(2).

CHAPTERS

`public static final org.olap4j.Axis.Standard` **`CHAPTERS`**

CHAPTERS axis, also known as AXIS(3).

SECTIONS

`public static final org.olap4j.Axis.Standard` **`SECTIONS`**

SECTIONS axis, also known as AXIS(4).

Methods

(continued from last page)

values

```
public final static Axis.Standard[] values()
```

valueOf

```
public static Axis.Standard valueOf(java.lang.String name)
```

axisOrdinal

```
public int axisOrdinal()
```

isFilter

```
public boolean isFilter()
```

getCaption

```
public java.lang.String getCaption(java.util.Locale locale)
```

org.olap4j
Class Axis.Factory

```
java.lang.Object
  |
  +-org.olap4j.Axis.Factory
```

public static class **Axis.Factory**
extends java.lang.Object

Container class for various Axis factory methods.

Constructor Summary

public	Axis.Factory()

Method Summary

static Axis	forOrdinal(int ordinal) Returns the axis with a given ordinal.

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructors

Axis.Factory

`public Axis.Factory()`

Methods

forOrdinal

`public static Axis forOrdinal(int ordinal)`

Returns the axis with a given ordinal.

For example, forOrdinal(0) returns the COLUMNS axis; forOrdinal(-1) returns the SLICER axis; forOrdinal(100) returns AXIS(100).

Parameters:

`ordinal` - Axis ordinal

Returns:

Axis whose ordinal is as given

org.olap4j
Interface Cell

public interface **Cell**
extends

Cell returned from a `CellSet`.

Method Summary

`java.sql.ResultSet`	`drillThrough()` Drills through from this cell to the underlying fact table data, and returns a `java.sql.ResultSet` of the results.
`CellSet`	`getCellSet()` Returns the `CellSet` that this Cell belongs to.
`java.util.List`	`getCoordinateList()` Returns the coordinates of this Cell in its `CellSetAxis`.
`double`	`getDoubleValue()` Returns the value of this cell as a `double` value.
`java.lang.String`	`getErrorText()` Returns the error message of this Cell, or null if the cell is not in error.
`java.lang.String`	`getFormattedValue()` Returns the value of this Cell, formatted according to the FORMAT_STRING property and using the numeric formatting tokens the current locale.
`int`	`getOrdinal()` Returns the ordinal of this Cell.
`java.lang.Object`	`getPropertyValue(Property property)` Returns the value of a given property for this Cell.
`java.lang.Object`	`getValue()` Returns the value of this Cell.
`boolean`	`isEmpty()` Returns whether this cell is empty.
`boolean`	`isError()` Returns whether an error occurred while evaluating this cell.
`boolean`	`isNull()` Returns whether the value of this cell is NULL.

Methods

getCellSet

```
public CellSet getCellSet()
```

(continued from last page)

Returns the `CellSet` that this Cell belongs to.

Returns:
CellSet, never null

getOrdinal

`public int **getOrdinal**()`

Returns the ordinal of this Cell.

The formula is the sequence, zero-based, which the cell would be visited in a raster-scan through all of the cells of this `CellSet`. The ordinal of the first cell is zero, and the ordinal of the last cell is the product of the lengths of the axes, minus 1. For example, if a result has 10 columns and 20 rows, then:

- (row 0, column 0) has ordinal 0,
- (row 0, column 1) has ordinal 1,
- (row 1, column 0) has ordinal 10,
- (row 19, column 9) has ordinal 199.

Returns:
Ordinal of this Cell

getCoordinateList

`public java.util.List **getCoordinateList**()`

Returns the coordinates of this Cell in its `CellSetAxis`.

This method is provided for convenience. It is equivalent to the following code:`getResult().ordinalToCoordinateList(getOrdinal())`

Returns:
Coordinates of this Cell

getPropertyValue

`public java.lang.Object **getPropertyValue**(Property property)`

Returns the value of a given property for this Cell.

The list of allowable properties may be obtained by calling `CellSet.getMetaData()` followed by `getCellProperties()`.

Every cell has certain system properties such as "VALUE" and "FORMAT_STRING" (the full list is described in the `Property.StandardCellProperty` enumeration), as well as extra properties defined by the query.

Parameters:
`property` - Property whose value to retrieve

Returns:
Value of the given property for this Cell; if the property is not set, returns null

isEmpty

`public boolean **isEmpty**()`

Returns whether this cell is empty.

(continued from last page)

Returns:
Whether this cell is empty.

isError

`public boolean isError()`

Returns whether an error occurred while evaluating this cell.

Returns:
Whether an error occurred while evaluating this cell.

isNull

`public boolean isNull()`

Returns whether the value of this cell is NULL.

Returns:
Whether the value of this cell is NULL.

getDoubleValue

```
public double getDoubleValue()
  throws OlapException
```

Returns the value of this cell as a `double` value.

Not all values can be represented as using the Java `double`, therefore for some providers, `getValue()` may return a more accurate result.

Returns:
The value of this cell; if the cell is null, the returns `0`

Throws:
`OlapException` - if this cell does not have a numeric value

getErrorText

`public java.lang.String getErrorText()`

Returns the error message of this Cell, or null if the cell is not in error.

If the cell is an error, the value will be an `OlapException`. (This value is returned, not thrown.)

Returns:
value of this Cell

getValue

`public java.lang.Object getValue()`

Returns the value of this Cell.

If the cell is an error, the value will be an `OlapException`. (This value is returned, not thrown.)

If the cell has a numeric value, returns an object which implements the `java.lang.Number` interface.

Returns:
value of this Cell

(continued from last page)

See Also:
`getDoubleValue()`

getFormattedValue

`public java.lang.String` **`getFormattedValue`**`()`

Returns the value of this Cell, formatted according to the FORMAT_STRING property and using the numeric formatting tokens the current locale.

The formatted value is never null. In particular, when the cell contains the MDX NULL value, `getValue()` will return the Java `null` value but this method will return the empty string `""`.

Returns:
Formatted value of this Cell

drillThrough

```
public java.sql.ResultSet drillThrough()
  throws OlapException
```

Drills through from this cell to the underlying fact table data, and returns a `java.sql.ResultSet` of the results.

If drill-through is not possible, returns null.

Returns:
result set of the fact rows underlying this Cell

Throws:
`OlapException` - if a database error occurs

org.olap4j
Interface CellSet

All Superinterfaces:
  OlapWrapper

public interface **CellSet**
extends java.sql.ResultSet, OlapWrapper

Result of executing an OLAP Statement.

An consists of a set of (typically two) axes, each populated with a sequence of members, and a collection of cells at the intersection of these axes.

Cell ordinals and coordinates

There are two ways to identify a particular cell: ordinal and coordinates. Suppose that there are `p` axes, and each axis `k` (`k` between 0 and `p - 1`) has `Uk` positions. There are `U` = `U0 * ... * Up - 1` cells in total. Then:

- A cell's `ordinal` is an integer between 0 and `U - 1`.
- A cell's `coordinates` are a list of `p` integers, indicating the cell's position on each axis. Each integer is between 0 and `Up-1`.

The ordinal number of a cell whose tuple ordinals are `(S0, S1, ... Sp-1)` is`i=0p-1 Si . Ei` where `E0 = 1` and `Ei = i=0p-1 Uk`

Fields inherited from interface `java.sql.ResultSet`

`CLOSE_CURSORS_AT_COMMIT`, `CONCUR_READ_ONLY`, `CONCUR_UPDATABLE`, `FETCH_FORWARD`, `FETCH_REVERSE`, `FETCH_UNKNOWN`, `HOLD_CURSORS_OVER_COMMIT`, `TYPE_FORWARD_ONLY`, `TYPE_SCROLL_INSENSITIVE`, `TYPE_SCROLL_SENSITIVE`

Method Summary

`int`	`coordinatesToOrdinal(java.util.List coordinates)` Converts a list of cell coordinates to a cell ordinal.
`java.util.List`	`getAxes()` Retrieves a list of CellSetAxis objects containing the result.
`Cell`	`getCell(int ordinal)` Returns the Cell at an ordinal.
`Cell`	`getCell(java.util.List coordinates)` Returns the Cell at a given set of coordinates.
`Cell`	`getCell(Position[] positions)` Returns the Cell at the intersection of a set of axis positions.
`CellSetAxis`	`getFilterAxis()` Retrieves the CellSetAxis representing the filter axis.
`CellSetMetaData`	`getMetaData()` Retrieves the description of this `CellSet`'s axes and cells.
`java.util.List`	`ordinalToCoordinates(int ordinal)` Converts a cell ordinal to a list of cell coordinates.

Methods inherited from interface `java.sql.ResultSet`

`absolute, afterLast, beforeFirst, cancelRowUpdates, clearWarnings, close, deleteRow, findColumn, first, getArray, getArray, getAsciiStream, getAsciiStream, getBigDecimal, getBigDecimal, getBigDecimal, getBigDecimal, getBinaryStream, getBinaryStream, getBlob, getBlob, getBoolean, getBoolean, getByte, getByte, getBytes, getBytes, getCharacterStream, getCharacterStream, getClob, getClob, getConcurrency, getCursorName, getDate, getDate, getDate, getDate, getDouble, getDouble, getFetchDirection, getFetchSize, getFloat, getFloat, getInt, getInt, getLong, getLong, getMetaData, getObject, getObject, getObject, getObject, getRef, getRef, getRow, getShort, getShort, getStatement, getString, getString, getTime, getTime, getTime, getTime, getTimestamp, getTimestamp, getTimestamp, getTimestamp, getType, getUnicodeStream, getUnicodeStream, getURL, getURL, getWarnings, insertRow, isAfterLast, isBeforeFirst, isFirst, isLast, last, moveToCurrentRow, moveToInsertRow, next, previous, refreshRow, relative, rowDeleted, rowInserted, rowUpdated, setFetchDirection, setFetchSize, updateArray, updateArray, updateAsciiStream, updateAsciiStream, updateBigDecimal, updateBigDecimal, updateBinaryStream, updateBinaryStream, updateBlob, updateBlob, updateBoolean, updateBoolean, updateByte, updateByte, updateBytes, updateBytes, updateCharacterStream, updateCharacterStream, updateClob, updateClob, updateDate, updateDate, updateDouble, updateDouble, updateFloat, updateFloat, updateInt, updateInt, updateLong, updateLong, updateNull, updateNull, updateObject, updateObject, updateObject, updateObject, updateRef, updateRef, updateRow, updateShort, updateShort, updateString, updateString, updateTime, updateTime, updateTimestamp, updateTimestamp, wasNull`

Methods inherited from interface `org.olap4j.OlapWrapper`

`isWrapperFor, unwrap`

Methods

getMetaData

```
public CellSetMetaData getMetaData()
  throws OlapException
```

Retrieves the description of this `CellSet`'s axes and cells.

Returns:
the description of this `CellSet`'s axes and cells

Throws:
`OlapException` - if a database access error occurs

getAxes

```
public java.util.List getAxes()
```

Retrieves a list of CellSetAxis objects containing the result.

The list contains axes according to their ordinal: 0 is the columns axis, 1 the rows axis, and so forth.

Returns:
list of CellSetAxis objects containing the result

(continued from last page)

getFilterAxis

`public CellSetAxis getFilterAxis()`

Retrieves the CellSetAxis representing the filter axis.

This axis always has one row, and contains one member for each dimension not included in any other axis. Some of these dimensions may have been explicitly mentioned in the WHERE clause of the MDX statement; others dimensions are represented by their default member.

Returns:
the filter axis

---

getCell

`public Cell getCell(java.util.List coordinates)`

Returns the Cell at a given set of coordinates.

Parameters:
`coordinates` - List of 0-based coordinates of the cell

Returns:
Cell

Throws:
`IndexOutOfBoundsException` - if coordinates are outside CellSet bounds

---

getCell

`public Cell getCell(int ordinal)`

Returns the Cell at an ordinal.

Equivalent to`getCell(ordinalToCoordinates(ordinal))`

Parameters:
`ordinal` - 0-based ordinal of the cell

Returns:
Cell

Throws:
`IndexOutOfBoundsException` - if ordinal lies outside CellSet bounds

---

getCell

`public Cell getCell(Position[] positions)`

(continued from last page)

Returns the Cell at the intersection of a set of axis positions.

Equivalent to

```
getCell(
     Arrays.asList(
         positions[0].ordinal(),
         positions[1].ordinal() [, ...]))
```

Parameters:
`positions` - Array of positions

Returns:
Cell

Throws:
`IllegalArgumentException` - if positions does not have the same number of members as the cell set has axes
`IndexOutOfBoundsException` - if positions lie outside CellSet bounds

ordinalToCoordinates

public java.util.List **ordinalToCoordinates**(int ordinal)

Converts a cell ordinal to a list of cell coordinates.

Parameters:
`ordinal` - Cell ordinal

Returns:
Cell coordinates

coordinatesToOrdinal

public int **coordinatesToOrdinal**(java.util.List coordinates)

Converts a list of cell coordinates to a cell ordinal.

Parameters:
`coordinates` - Cell coordinates

Returns:
Cell ordinal

org.olap4j
Interface CellSetAxis

public interface **CellSetAxis**
extends java.lang.Iterable

Axis of a CellSet.

A cell set has the same number of axes as the MDX statement which was executed to produce it. For example, a typical cell set, resulting from an MDX query with COLUMNS and ROWS expressions is two-dimensional, and therefore has two axes.

Each axis is an ordered collection of members or tuples. Each member or tuple on an axis is called a `Position`.

The positions on the cell set axis can be accessed sequentially or random-access. Use the `getPositions()` method to return a list for random access, or the `iterator()` method to obtain an iterator for sequential access.

Method Summary

`CellSetAxisMetaData`	`getAxisMetaData()` Returns a description of the type (e.g.
`Axis`	`getAxisOrdinal()` Returns the axis ordinal of this `CellSetAxis`.
`CellSet`	`getCellSet()` Returns the `CellSet` which this `CellSetAxis` belongs to.
`int`	`getPositionCount()` Returns the number of positions on this CellSetAxis.
`java.util.List`	`getPositions()` Returns a list of `Position` objects on this CellSetAxis.
`java.util.ListIterator`	`iterator()` Opens an iterator over the positions on this CellSetAxis.

Methods inherited from interface `java.lang.Iterable`

`iterator`

Methods

getAxisOrdinal

```
public Axis getAxisOrdinal()
```

Returns the axis ordinal of this `CellSetAxis`.

The first axis in a CellSet will return `COLUMNS`, the second `ROWS`, and so forth, as described by the `axisOrdinal()` method of the `Axis` enumeration.

Returns:
the ordinal of this axis

getCellSet

`public CellSet getCellSet()`

Returns the `CellSet` which this `CellSetAxis` belongs to.

Returns:
the CellSet

getAxisMetaData

`public CellSetAxisMetaData getAxisMetaData()`

Returns a description of the type (e.g. `ROWS`) of this axis, and the hierarchies and properties which will be found on it.

The result is identical to evaluating`getCellSet().getMetaData().getSlicerAxisMetaData()`for a filter axis, and `getCellSet().getMetaData().getAxesMetaData().get(getAxisOrdinal().axisOrdinal())`for other axes.

Returns:
metadata describing this CellSetAxis

getPositions

`public java.util.List getPositions()`

Returns a list of `Position` objects on this CellSetAxis.

Returns:
List of positions on this axis (never null)

getPositionCount

`public int getPositionCount()`

Returns the number of positions on this CellSetAxis.

This method can be called at any time. In particular, it is not necessary to complete an iteration through all positions before calling this method.

The number of positions on an axis is important when computing the ordinal of a cell.

Returns:
the number of positions

iterator

`public java.util.ListIterator iterator()`

(continued from last page)

Opens an iterator over the positions on this CellSetAxis.

If this axis has very many positions, this method may be more efficient than `getPositions()`.

This method allows CellSetAxis to implement the `java.lang.Iterable` interface, so one might use it in a foreach construct, for example:

```
CellSet cellSet;
for (Position rowPos : cellSet.getAxes().get(0)) {
    for (Position colPos : cellSet.getAxes().get(1)) {
         Cell cell = cellSet.getCell(colPos, rowPos);
         ....
    }
}
```

Returns:

iterator over the collection of positions

org.olap4j
Interface CellSetAxisMetaData

public interface **CellSetAxisMetaData**
extends

Description of structure of a particular axis of an `CellSet`.

For example, in the MDX statement

```
SELECT
  {[Measures].Members} ON COLUMNS,
  CrossJoin([Store].Members, [Gender].Children)
  DIMENSION PROPERTIES
     MEMBER_ORDINAL,
     MEMBER_UNIQUE_NAME,
     DISPLAY_INFO ON ROWS
FROM [Sales]
```

the ROWS axis is described by the following metadata:

Attribute	Value
hierarchies	{[Store], [Gender]}
properties	{MEMBER_ORDINAL, MEMBER_UNIQUE_NAME, DISPLAY_INFO}

Method Summary

`Axis`	`getAxisOrdinal()` Returns the definition of the axis.
`java.util.List`	`getHierarchies()` Returns the hierarchies which are mapped onto this axis.
`java.util.List`	`getProperties()` Returns the member properties which are returned on this axis.

Methods

getAxisOrdinal

`public Axis getAxisOrdinal()`

Returns the definition of the axis. Typical values are (`FILTER`, `COLUMNS`, `ROWS`, and so forth.)

Returns:
the Axis

(continued from last page)

getHierarchies

```
public java.util.List getHierarchies()
```

Returns the hierarchies which are mapped onto this axis.

Returns:
list of hierarchies on this Axis

getProperties

```
public java.util.List getProperties()
```

Returns the member properties which are returned on this axis.

This method does not return a `NamedList` because the names of the properties are not necessarily unique; for example, there might be two hierarchies on the axis, each of which returns the DISPLAY_INFO property.

Returns:
list of member properties on this Axis

org.olap4j
Interface CellSetMetaData

All Superinterfaces:

OlapWrapper

---

public interface **CellSetMetaData**
extends java.sql.ResultSetMetaData, OlapWrapper

An object that can be used to get information about the axes and cells in a `CellSet` object.

The following code fragment creates the `CellSet` object cs, creates the `CellSetMetaData` object csmd, and uses csmd to find out how many axes cs has and the name of the cube.

```
CellSet cs = stmt.executeOlapQuery(
    "SELECT {[Measures].[Unit Sales] ON COLUMNS,\n" +
    "   Crossjoin([Time].Children, [Store].Children) ON ROWS\n" +
    "FROM [Sales]");
CellSetMetaData csmd = cs.getMetaData();
int numberOfAxes = csmd.getAxesMetaData().size();
String cubeName = csmd.getCube().getName();
```

Fields inherited from interface `java.sql.ResultSetMetaData`

`columnNoNulls`, `columnNullable`, `columnNullableUnknown`

Method Summary

`NamedList`	`getAxesMetaData()` Returns a list of CellSetAxisMetaData describing each result axis.
`NamedList`	`getCellProperties()` Returns a list of Property objects which each Cell may have.
`Cube`	`getCube()` Returns the Cube which was referenced in this statement.
`CellSetAxisMetaData`	`getFilterAxisMetaData()` Returns a CellSetAxisMetaData describing the filter axis.

Methods inherited from interface `java.sql.ResultSetMetaData`

`getCatalogName, getColumnClassName, getColumnCount, getColumnDisplaySize, getColumnLabel, getColumnName, getColumnType, getColumnTypeName, getPrecision, getScale, getSchemaName, getTableName, isAutoIncrement, isCaseSensitive, isCurrency, isDefinitelyWritable, isNullable, isReadOnly, isSearchable, isSigned, isWritable`

Methods inherited from interface `org.olap4j.OlapWrapper`

`isWrapperFor`, `unwrap`

Methods

getCellProperties

`public NamedList getCellProperties()`

Returns a list of Property objects which each Cell may have.

Returns:
list of cell properties

getCube

`public Cube getCube()`

Returns the Cube which was referenced in this statement.

Returns:
cube referenced in this statement

getAxesMetaData

`public NamedList getAxesMetaData()`

Returns a list of CellSetAxisMetaData describing each result axis.

Returns:
list of metadata describing each result axis

getFilterAxisMetaData

`public CellSetAxisMetaData getFilterAxisMetaData()`

Returns a CellSetAxisMetaData describing the filter axis. Never returns null; if the MDX statement contains no WHERE clause, the description of the filter contains no hierarchies.

Returns:
metadata describing filter axis

org.olap4j
Interface OlapConnection

All Superinterfaces:
OlapWrapper

public interface **OlapConnection**
extends java.sql.Connection, OlapWrapper

Connection to an OLAP server.

Fields inherited from interface `java.sql.Connection`
`TRANSACTION_NONE`, `TRANSACTION_READ_COMMITTED`, `TRANSACTION_READ_UNCOMMITTED`, `TRANSACTION_REPEATABLE_READ`, `TRANSACTION_SERIALIZABLE`

Method Summary

`OlapStatement`	`createStatement()`
`NamedList`	`getCatalogs()` Returns a list of `Catalog` objects which belong to this connection's OLAP server.
`java.util.Locale`	`getLocale()` Returns this connection's locale.
`OlapDatabaseMetaData`	`getMetaData()`
`MdxParserFactory`	`getParserFactory()` Returns the factory used to create MDX parsers in this connection.
`java.lang.String`	`getRoleName()` Returns the name of the role in which this connection executes queries.
`Schema`	`getSchema()` Returns the current `Schema` of this connection.
`PreparedOlapStatement`	`prepareOlapStatement(java.lang.String mdx)` Creates a prepared OLAP Statement.
`void`	`setLocale(java.util.Locale locale)` Sets the current locale of this connection.
`void`	`setRoleName(java.lang.String roleName)` Sets the name of the role in which this connection executes queries.

Methods inherited from interface `java.sql.Connection`

clearWarnings, close, commit, createStatement, createStatement, createStatement, getAutoCommit, getCatalog, getHoldability, getMetaData, getTransactionIsolation, getTypeMap, getWarnings, isClosed, isReadOnly, nativeSQL, prepareCall, prepareCall, prepareCall, prepareStatement, prepareStatement, prepareStatement, prepareStatement, prepareStatement, prepareStatement, releaseSavepoint, rollback, rollback, setAutoCommit, setCatalog, setHoldability, setReadOnly, setSavepoint, setSavepoint, setTransactionIsolation, setTypeMap

Methods inherited from interface org.olap4j.OlapWrapper

isWrapperFor, unwrap

Methods

getMetaData

```
public OlapDatabaseMetaData getMetaData()
  throws OlapException
```

Throws:

OlapException - if database error occurs

prepareOlapStatement

```
public PreparedOlapStatement prepareOlapStatement(java.lang.String mdx)
  throws OlapException
```

Creates a prepared OLAP Statement.

This method is the equivalent, for OLAP, of the `Connection.prepareStatement(java.lang.String)` JDBC method.

Parameters:

`mdx` - MDX query string

Returns:

prepared statement

Throws:

OlapException - if database error occurs

getParserFactory

```
public MdxParserFactory getParserFactory()
```

Returns the factory used to create MDX parsers in this connection.

Returns:

MDX parser factory

createStatement

```
public OlapStatement createStatement()
  throws OlapException
```

(continued from last page)

Throws:
`OlapException` - if database error occurs

getSchema

```
public Schema getSchema()
  throws OlapException
```

Returns the current `Schema` of this connection.

Returns:
current Schema

Throws:
`OlapException` - if database error occurs

getCatalogs

```
public NamedList getCatalogs()
```

Returns a list of `Catalog` objects which belong to this connection's OLAP server.

The caller should assume that the list is immutable; if the caller modifies the list, behavior is undefined.

Returns:
List of Catalogs in this connection's OLAP server

See Also:
`getCatalogs()`

setLocale

```
public void setLocale(java.util.Locale locale)
```

Sets the current locale of this connection. The value must not be null.

If the locale is not set, the JDK's current locale is used (see `Locale.getDefault()`).

Most drivers support a `Locale` connect-string property.

Parameters:
`locale` - Locale

getLocale

```
public java.util.Locale getLocale()
```

Returns this connection's locale. The value is never null.

Returns:
locale of this connection

setRoleName

```
public void setRoleName(java.lang.String roleName)
  throws OlapException
```

(continued from last page)

Sets the name of the role in which this connection executes queries. If the name of the role is null, the connection reverts to the default access control context.

Parameters:

`roleName` - Name of role

Throws:

`OlapException` - if role name is invalid

getRoleName

```
public java.lang.String getRoleName()
```

Returns the name of the role in which this connection executes queries.

Returns:

name of the role in which this connection executes queries

org.olap4j
Interface OlapDatabaseMetaData

All Superinterfaces:
OlapWrapper

public interface **OlapDatabaseMetaData**
extends java.sql.DatabaseMetaData, OlapWrapper

Information about an OLAP database.

Methods are provided to query the metadata catalog of the database. There is a method for each metadata class, and each method takes zero or more parameters to qualify the instances should be returned, and returns a JDBC `java.sql.ResultSet`.

For example, `getCubes(String, String, String)` returns the description of a cube.

Fields inherited from interface `java.sql.DatabaseMetaData`

`attributeNoNulls`, `attributeNullable`, `attributeNullableUnknown`, `bestRowNotPseudo`, `bestRowPseudo`, `bestRowSession`, `bestRowTemporary`, `bestRowTransaction`, `bestRowUnknown`, `columnNoNulls`, `columnNullable`, `columnNullableUnknown`, `importedKeyCascade`, `importedKeyInitiallyDeferred`, `importedKeyInitiallyImmediate`, `importedKeyNoAction`, `importedKeyNotDeferrable`, `importedKeyRestrict`, `importedKeySetDefault`, `importedKeySetNull`, `procedureColumnIn`, `procedureColumnInOut`, `procedureColumnOut`, `procedureColumnResult`, `procedureColumnReturn`, `procedureColumnUnknown`, `procedureNoNulls`, `procedureNoResult`, `procedureNullable`, `procedureNullableUnknown`, `procedureResultUnknown`, `procedureReturnsResult`, `sqlStateSQL99`, `sqlStateXOpen`, `tableIndexClustered`, `tableIndexHashed`, `tableIndexOther`, `tableIndexStatistic`, `typeNoNulls`, `typeNullable`, `typeNullableUnknown`, `typePredBasic`, `typePredChar`, `typePredNone`, `typeSearchable`, `versionColumnNotPseudo`, `versionColumnPseudo`, `versionColumnUnknown`

Method Summary

`java.sql.ResultSet`	`getActions(java.lang.String catalog, java.lang.String schemaPattern, java.lang.String cubeNamePattern, java.lang.String actionNamePattern)` Retrieves a result set describing the Actions in this database.
`OlapConnection`	`getConnection()`
`java.sql.ResultSet`	`getCubes(java.lang.String catalog, java.lang.String schemaPattern, java.lang.String cubeNamePattern)` Retrieves a result set describing the Cubes in this database.
`java.sql.ResultSet`	`getDatabaseProperties(java.lang.String dataSourceName, java.lang.String propertyNamePattern)` Retrieves a list of the standard and provider-specific properties supported by an olap4j provider.
`java.sql.ResultSet`	`getDatasources()` Retrives a list of olap4j data sources that are available on the server.
`java.sql.ResultSet`	`getDimensions(java.lang.String catalog, java.lang.String schemaPattern, java.lang.String cubeNamePattern, java.lang.String dimensionNamePattern)` Retrieves a result set describing the shared and private Dimensions in this database.

java.sql.ResultSet	getHierarchies(java.lang.String catalog, java.lang.String schemaPattern, java.lang.String cubeNamePattern, java.lang.String dimensionUniqueName, java.lang.String hierarchyNamePattern) Retrieves a result set describing the Hierarchies in this database.
java.sql.ResultSet	getLevels(java.lang.String catalog, java.lang.String schemaPattern, java.lang.String cubeNamePattern, java.lang.String dimensionUniqueName, java.lang.String hierarchyUniqueName, java.lang.String levelNamePattern) Retrieves a result set describing the Levels in this database.
java.sql.ResultSet	getLiterals() Retrieves a list of information on supported literals, including data types and values.
java.lang.String	getMdxKeywords() Retrieves a comma-separated list of all of this database's MDX keywords.
java.sql.ResultSet	getMeasures(java.lang.String catalog, java.lang.String schemaPattern, java.lang.String cubeNamePattern, java.lang.String measureNamePattern, java.lang.String measureUniqueName) Retrieves a result set describing the Measures in this database.
java.sql.ResultSet	getMembers(java.lang.String catalog, java.lang.String schemaPattern, java.lang.String cubeNamePattern, java.lang.String dimensionUniqueName, java.lang.String hierarchyUniqueName, java.lang.String levelUniqueName, java.lang.String memberUniqueName, java.util.Set treeOps) Retrieves a result set describing the Members in this database.
java.sql.ResultSet	getOlapFunctions(java.lang.String functionNamePattern) Retrieves a result set describing the Functions available to client applications connected to the database.
java.sql.ResultSet	getProperties(java.lang.String catalog, java.lang.String schemaPattern, java.lang.String cubeNamePattern, java.lang.String dimensionUniqueName, java.lang.String hierarchyUniqueName, java.lang.String levelUniqueName, java.lang.String memberUniqueName, java.lang.String propertyNamePattern) Retrieves a result set describing member and cell Properties.
java.sql.ResultSet	getSets(java.lang.String catalog, java.lang.String schemaPattern, java.lang.String cubeNamePattern, java.lang.String setNamePattern) Retrieves a result set describing the named Sets in this database.

Methods inherited from interface `java.sql.DatabaseMetaData` allProceduresAreCallable, allTablesAreSelectable, dataDefinitionCausesTransactionCommit, dataDefinitionIgnoredInTransactions, deletesAreDetected, doesMaxRowSizeIncludeBlobs, getAttributes, getBestRowIdentifier, getCatalogs, getCatalogSeparator, getCatalogTerm, getColumnPrivileges, getColumns, getConnection, getCrossReference, getDatabaseMajorVersion, getDatabaseMinorVersion, getDatabaseProductName, getDatabaseProductVersion, getDefaultTransactionIsolation, getDriverMajorVersion, getDriverMinorVersion, getDriverName, getDriverVersion, getExportedKeys, getExtraNameCharacters, getIdentifierQuoteString, getImportedKeys, getIndexInfo, getJDBCMajorVersion, getJDBCMinorVersion, getMaxBinaryLiteralLength, getMaxCatalogNameLength, getMaxCharLiteralLength, getMaxColumnNameLength, getMaxColumnsInGroupBy, getMaxColumnsInIndex, getMaxColumnsInOrderBy, getMaxColumnsInSelect, getMaxColumnsInTable, getMaxConnections, getMaxCursorNameLength, getMaxIndexLength, getMaxProcedureNameLength, getMaxRowSize, getMaxSchemaNameLength, getMaxStatementLength, getMaxStatements, getMaxTableNameLength, getMaxTablesInSelect, getMaxUserNameLength, getNumericFunctions, getPrimaryKeys, getProcedureColumns, getProcedures, getProcedureTerm, getResultSetHoldability, getSchemas, getSchemaTerm, getSearchStringEscape, getSQLKeywords, getSQLStateType, getStringFunctions, getSuperTables, getSuperTypes, getSystemFunctions, getTablePrivileges, getTables, getTableTypes, getTimeDateFunctions, getTypeInfo, getUDTs, getURL, getUserName, getVersionColumns, insertsAreDetected, isCatalogAtStart, isReadOnly, locatorsUpdateCopy, nullPlusNonNullIsNull, nullsAreSortedAtEnd, nullsAreSortedAtStart, nullsAreSortedHigh, nullsAreSortedLow, othersDeletesAreVisible, othersInsertsAreVisible, othersUpdatesAreVisible, ownDeletesAreVisible, ownInsertsAreVisible, ownUpdatesAreVisible, storesLowerCaseIdentifiers, storesLowerCaseQuotedIdentifiers, storesMixedCaseIdentifiers, storesMixedCaseQuotedIdentifiers, storesUpperCaseIdentifiers, storesUpperCaseQuotedIdentifiers, supportsAlterTableWithAddColumn, supportsAlterTableWithDropColumn, supportsANSI92EntryLevelSQL, supportsANSI92FullSQL, supportsANSI92IntermediateSQL, supportsBatchUpdates, supportsCatalogsInDataManipulation, supportsCatalogsInIndexDefinitions, supportsCatalogsInPrivilegeDefinitions, supportsCatalogsInProcedureCalls, supportsCatalogsInTableDefinitions, supportsColumnAliasing, supportsConvert, supportsConvert, supportsCoreSQLGrammar, supportsCorrelatedSubqueries, supportsDataDefinitionAndDataManipulationTransactions, supportsDataManipulationTransactionsOnly, supportsDifferentTableCorrelationNames, supportsExpressionsInOrderBy, supportsExtendedSQLGrammar, supportsFullOuterJoins, supportsGetGeneratedKeys, supportsGroupBy, supportsGroupByBeyondSelect, supportsGroupByUnrelated, supportsIntegrityEnhancementFacility, supportsLikeEscapeClause, supportsLimitedOuterJoins, supportsMinimumSQLGrammar, supportsMixedCaseIdentifiers, supportsMixedCaseQuotedIdentifiers, supportsMultipleOpenResults, supportsMultipleResultSets, supportsMultipleTransactions, supportsNamedParameters, supportsNonNullableColumns, supportsOpenCursorsAcrossCommit, supportsOpenCursorsAcrossRollback, supportsOpenStatementsAcrossCommit, supportsOpenStatementsAcrossRollback, supportsOrderByUnrelated, supportsOuterJoins, supportsPositionedDelete, supportsPositionedUpdate, supportsResultSetConcurrency, supportsResultSetHoldability, supportsResultSetType, supportsSavepoints, supportsSchemasInDataManipulation, supportsSchemasInIndexDefinitions, supportsSchemasInPrivilegeDefinitions, supportsSchemasInProcedureCalls, supportsSchemasInTableDefinitions, supportsSelectForUpdate, supportsStatementPooling, supportsStoredProcedures, supportsSubqueriesInComparisons, supportsSubqueriesInExists, supportsSubqueriesInIns, supportsSubqueriesInQuantifieds, supportsTableCorrelationNames, supportsTransactionIsolationLevel, supportsTransactions, supportsUnion, supportsUnionAll, updatesAreDetected, usesLocalFilePerTable, usesLocalFiles

Methods inherited from interface org.olap4j.OlapWrapper isWrapperFor, unwrap

Methods

getConnection

```
public OlapConnection getConnection()
  throws java.sql.SQLException
```

getActions

```
public java.sql.ResultSet getActions(java.lang.String catalog,
         java.lang.String schemaPattern,
         java.lang.String cubeNamePattern,
         java.lang.String actionNamePattern)
  throws OlapException
```

Retrieves a result set describing the Actions in this database.

Specification as for XML/A MDSCHEMA_ACTIONS schema rowset.

Each action description has the following columns:

- **SCHEMA_NAME** String (may be `null`) => The name of the schema to which this action belongs.
- **CUBE_NAME** String => The name of the cube to which this action belongs.
- **ACTION_NAME** String => The name of the action.
- **COORDINATE** String => null
- **COORDINATE_TYPE** int => null

Parameters:

`catalog` - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; `null` means that the catalog name should not be used to narrow the search
`schemaPattern` - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; `null` means that the schema name should not be used to narrow the search
`cubeNamePattern` - a cube name pattern; must match the cube name as it is stored in the database; "" retrieves those without a cube (such as shared dimensions); `null` means that the cube name should not be used to narrow the search
`actionNamePattern` - an action name pattern; must match the action name as it is stored in the database; `null` means that the action name should not be used to narrow the search

Returns:

a `ResultSet` object in which each row is an action description

Throws:

`OlapException` - if a database access error occurs

See Also:

`DatabaseMetaData.getSearchStringEscape()`

getDatasources

```
public java.sql.ResultSet getDatasources()
  throws OlapException
```

(continued from last page)

Retrives a list of olap4j data sources that are available on the server.

Specification as for XML/A DISCOVER_DATASOURCES schema rowset.

1. **DATA_SOURCE_NAME** String => The name of the data source, such as FoodMart 2000.
2. **DATA_SOURCE_DESCRIPTION** String => A description of the data source, as entered by the publisher. (may be `null`)
3. **URL** String => The unique path that shows where to invoke the XML for Analysis methods for that data source. (may be `null`)
4. **DATA_SOURCE_INFO** String => A string containing any additional information required to connect to the data source. This can include the Initial Catalog property or other information for the provider.
 Example: "Provider=MSOLAP;Data Source=Local;" (may be `null`)
5. **PROVIDER_NAME** String => The name of the provider behind the data source.
 Example: "MSDASQL" (may be `null`)
6. **PROVIDER_TYPE** EnumerationArray => The types of data supported by the provider. May include one or more of the following types. Example follows this table.
 TDP: tabular data provider.
 MDP: multidimensional data provider.
 DMP: data mining provider. A DMP provider implements the OLE DB for Data Mining specification.
7. **AUTHENTICATION_MODE** EnumString => Specification of what type of security mode the data source uses. Values can be one of the following:
 Unauthenticated: no user ID or password needs to be sent.
 Authenticated: User ID and Password must be included in the information required for the connection.
 Integrated: the data source uses the underlying security to determine authorization, such as Integrated Security provided by Microsoft Internet Information Services (IIS).

Returns:

a `ResultSet` object in which each row is a datasource description

Throws:

`OlapException` - if a database access error occurs

getLiterals

```
public java.sql.ResultSet getLiterals()
  throws OlapException
```

Retrieves a list of information on supported literals, including data types and values.

Specification as for XML/A DISCOVER_LITERALS schema rowset.

1. **LITERAL_NAME** String => The name of the literal described in the row.
 Example: DBLITERAL_LIKE_PERCENT
2. **LITERAL_VALUE** String (may be `null`) => Contains the actual literal value.
 Example, if LiteralName is DBLITERAL_LIKE_PERCENT and the percent character (%) is used to match zero or more characters in a LIKE clause, this column's value would be "%".
3. **LITERAL_INVALID_CHARS** String (may be `null`) => The characters, in the literal, that are not valid.
 For example, if table names can contain anything other than a numeric character, this string would be "0123456789".
4. **LITERAL_INVALID_STARTING_CHARS** String (may be `null`) => The characters that are not valid as the first character of the literal. If the literal can start with any valid character, this is null.
5. **LITERAL_MAX_LENGTH** int (may be `null`) => The maximum number of characters in the literal. If there is no maximum or the maximum is unknown, the value is -1.

Returns:

a `ResultSet` object in which each row is a literal description

Throws:

`OlapException` - if a database access error occurs

getDatabaseProperties

```
public java.sql.ResultSet getDatabaseProperties(java.lang.String dataSourceName,
         java.lang.String propertyNamePattern)
  throws OlapException
```

Retrieves a list of the standard and provider-specific properties supported by an olap4j provider. Properties that are not supported by a provider are not listed in the return result set.

Specification as for XML/A DISCOVER_PROPERTIES schema rowset.

Not to be confused with `getProperties(String, String, String, String, String, String, String, String)`.

1. **PROPERTY_NAME** String => The name of the property.
2. **PROPERTY_DESCRIPTION** String => A localizable text description of the property.
3. **PROPERTY_TYPE** String => The XML data type of the property.
4. **PROPERTY_ACCESS_TYPE** EnumString => Access for the property. The value can be Read, Write, or ReadWrite.
5. **IS_REQUIRED** Boolean => True if a property is required, false if it is not required.
6. **PROPERTY_VALUE** String => The current value of the property.

Parameters:

`dataSourceName` - Name of data source

`propertyNamePattern` - an property name pattern; must match the property name as it is stored in the database; `null` means that the property name should not be used to narrow the search

Returns:

a `ResultSet` object in which each row is a the description of a database property

Throws:

`OlapException` - if a database access error occurs

See Also:

`DatabaseMetaData.getSearchStringEscape()`

getProperties

```
public java.sql.ResultSet getProperties(java.lang.String catalog,
         java.lang.String schemaPattern,
         java.lang.String cubeNamePattern,
         java.lang.String dimensionUniqueName,
         java.lang.String hierarchyUniqueName,
         java.lang.String levelUniqueName,
         java.lang.String memberUniqueName,
         java.lang.String propertyNamePattern)
  throws OlapException
```

(continued from last page)

Retrieves a result set describing member and cell Properties.

Specification as for XML/A MDSCHEMA_PROPERTIES schema rowset.

Not to be confused with `getDatabaseProperties(String, String)`.

- **CATALOG_NAME** String (may be `null`) => The name of the database.
- **SCHEMA_NAME** String (may be `null`) => The name of the schema to which this property belongs.
- **CUBE_NAME** String => The name of the cube.
- **DIMENSION_UNIQUE_NAME** String => The unique name of the dimension.
- **HIERARCHY_UNIQUE_NAME** String => The unique name of the hierarchy.
- **LEVEL_UNIQUE_NAME** String => The unique name of the level to which this property belongs.
- **MEMBER_UNIQUE_NAME** String (may be `null`) => The unique name of the member to which the property belongs.
- **PROPERTY_NAME** String => Name of the property.
- **PROPERTY_CAPTION** String => A label or caption associated with the property, used primarily for display purposes.
- **PROPERTY_TYPE** Short => A bitmap that specifies the type of the property
- **DATA_TYPE** UnsignedShort => Data type of the property.
- **PROPERTY_CONTENT_TYPE** Short (may be `null`) => The type of the property.
- **DESCRIPTION** String (may be `null`) => A human-readable description of the measure.

Parameters:

`catalog` - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; `null` means that the catalog name should not be used to narrow the search
`schemaPattern` - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; `null` means that the schema name should not be used to narrow the search
`cubeNamePattern` - a cube name pattern; must match the cube name as it is stored in the database; "" retrieves those without a cube; `null` means that the cube name should not be used to narrow the search
`dimensionUniqueName` - unique name of a dimension (not a pattern); must match the dimension name as it is stored in the database; `null` means that the dimension name should not be used to narrow the search
`hierarchyUniqueName` - unique name of a hierarchy (not a pattern); must match the hierarchy name as it is stored in the database; `null` means that the hierarchy name should not be used to narrow the search
`levelUniqueName` - unique name of a level (not a pattern); must match the level name as it is stored in the database; `null` means that the level name should not be used to narrow the search
`memberUniqueName` - unique name of member (not a pattern); `null` means that the member unique name should not be used to narrow the search
`propertyNamePattern` - a property name pattern; must match the property name as it is stored in the database; `null` means that the property name should not be used to narrow the search

Returns:

a `ResultSet` object in which each row is a description of a member or cell property

Throws:

`OlapException` - if a database access error occurs

See Also:

`DatabaseMetaData.getSearchStringEscape()`
`Property`

getMdxKeywords

```
public java.lang.String getMdxKeywords()
  throws OlapException
```

Retrieves a comma-separated list of all of this database's MDX keywords.

Returns:

the list of this database's MDX keywords

Throws:

`OlapException` - if a database access error occurs

getCubes

```
public java.sql.ResultSet getCubes(java.lang.String catalog,
         java.lang.String schemaPattern,
         java.lang.String cubeNamePattern)
  throws OlapException
```

Retrieves a result set describing the Cubes in this database.

Specification as for XML/A MDSCHEMA_CUBES schema rowset.

Each cube description has the following columns:

1. **CATALOG_NAME** String (may be `null`) => The name of the catalog to which this cube belongs.
2. **SCHEMA_NAME** String (may be `null`) => The name of the schema to which this cube belongs.
3. **CUBE_NAME** String => Name of the cube.
4. **CUBE_TYPE** String => Cube type.
5. **CUBE_GUID** UUID (may be `null`) => Cube type.
6. **CREATED_ON** Timestamp (may be `null`) => Date and time of cube creation.
7. **LAST_SCHEMA_UPDATE** Timestamp (may be `null`) => Date and time of last schema update.
8. **SCHEMA_UPDATED_BY** String (may be `null`) => User ID of the person who last updated the schema.
9. **LAST_DATA_UPDATE** Timestamp (may be `null`) => Date and time of last data update.
10. **DATA_UPDATED_BY** String (may be `null`) => User ID of the person who last updated the data.
11. **IS_DRILLTHROUGH_ENABLED** boolean => Describes whether DRILLTHROUGH can be performed on the members of a cube
12. **IS_WRITE_ENABLED** boolean => Describes whether a cube is write-enabled
13. **IS_LINKABLE** boolean => Describes whether a cube can be used in a linked cube
14. **IS_SQL_ENABLED** boolean => Describes whether or not SQL can be used on the cube
15. **DESCRIPTION** String (may be `null`) => A user-friendly description of the dimension.

Parameters:

`catalog` - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; `null` means that the catalog name should not be used to narrow the search
`schemaPattern` - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; `null` means that the schema name should not be used to narrow the search
`cubeNamePattern` - a cube name pattern; must match the cube name as it is stored in the database; `null` means that the cube name should not be used to narrow the search

Returns:

`ResultSet` in which each row is a cube description

Throws:

`OlapException` - if a database access error occurs

See Also:

`DatabaseMetaData.getSearchStringEscape()`
`Cube`

getDimensions

```
public java.sql.ResultSet getDimensions(java.lang.String catalog,
         java.lang.String schemaPattern,
         java.lang.String cubeNamePattern,
         java.lang.String dimensionNamePattern)
  throws OlapException
```

(continued from last page)

Retrieves a result set describing the shared and private Dimensions in this database.

Specification as for XML/A MDSCHEMA_DIMENSIONS schema rowset.

Each dimension description has the following columns:

1. **CATALOG_NAME** String (may be `null`) => The name of the database.
2. **SCHEMA_NAME** String (may be `null`) => Not supported.
3. **CUBE_NAME** String => The name of the cube.
4. **DIMENSION_NAME** String => The name of the dimension.
5. **DIMENSION_UNIQUE_NAME** String => The unique name of the dimension.
6. **DIMENSION_GUID** String (may be `null`) => Not supported.
7. **DIMENSION_CAPTION** String => The caption of the dimension.
8. **DIMENSION_ORDINAL** int => The position of the dimension within the cube.
9. **DIMENSION_TYPE** Short => The type of the dimension.
10. **DIMENSION_CARDINALITY** int => The number of members in the key attribute.
11. **DEFAULT_HIERARCHY** String => A hierarchy from the dimension. Preserved for backwards compatibility.
12. **DESCRIPTION** String (may be `null`) => A user-friendly description of the dimension.
13. **IS_VIRTUAL** boolean (may be `null`) => Always FALSE.
14. **IS_READWRITE** boolean (may be `null`) => A Boolean that indicates whether the dimension is write-enabled.
15. **DIMENSION_UNIQUE_SETTINGS** int (may be `null`) => A bitmap that specifies which columns contain unique values if the dimension contains only members with unique names.
16. **DIMENSION_MASTER_UNIQUE_NAME** String (may be `null`) => Always NULL.
17. **DIMENSION_IS_VISIBLE** boolean (may be `null`) => Always TRUE.

Parameters:

`catalog` - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; `null` means that the catalog name should not be used to narrow the search
`schemaPattern` - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; `null` means that the schema name should not be used to narrow the search
`cubeNamePattern` - a cube name pattern; must match the cube name as it is stored in the database; "" retrieves those without a cube (such as shared dimensions); `null` means that the cube name should not be used to narrow the search
`dimensionNamePattern` - a dimension name pattern; must match the dimension name as it is stored in the database; `null` means that the dimension name should not be used to narrow the search

Returns:

a `ResultSet` object in which each row is a dimension description

Throws:

`OlapException` - if a database access error occurs

See Also:

`DatabaseMetaData.getSearchStringEscape()`
`Dimension`

getOlapFunctions

```
public java.sql.ResultSet getOlapFunctions(java.lang.String functionNamePattern)
  throws OlapException
```

(continued from last page)

Retrieves a result set describing the Functions available to client applications connected to the database.

Specification as for XML/A MDSCHEMA_FUNCTIONS schema rowset.

Each function description has the following columns:

- **FUNCTION_NAME** String => The name of the function.
- **DESCRIPTION** String (may be `null`) => A description of the function.
- **PARAMETER_LIST** String (may be `null`) => A comma delimited list of parameters.
- **RETURN_TYPE** int => The VARTYPE of the return data type of the function.
- **ORIGIN** int => The origin of the function: 1 for MDX functions. 2 for user-defined functions.
- **INTERFACE_NAME** String => The name of the interface for user-defined functions
- **LIBRARY_NAME** String (may be `null`) => The name of the type library for user-defined functions. NULL for MDX functions.
- **CAPTION** String (may be `null`) => The display caption for the function.

Parameters:

`functionNamePattern` - a function name pattern; must match the function name as it is stored in the database; `null` means that the function name should not be used to narrow the search

Returns:

a `ResultSet` object in which each row is a function description

Throws:

`OlapException` - if a database access error occurs

See Also:

`getFunctions(String, String, String)`
`DatabaseMetaData.getSearchStringEscape()`

getHierarchies

```
public java.sql.ResultSet getHierarchies(java.lang.String catalog,
         java.lang.String schemaPattern,
         java.lang.String cubeNamePattern,
         java.lang.String dimensionUniqueName,
         java.lang.String hierarchyNamePattern)
  throws OlapException
```

(continued from last page)

Retrieves a result set describing the Hierarchies in this database.

Specification as for XML/A MDSCHEMA_HIERARCHIES schema rowset.

Each hierarchy description has the following columns:

- **CATALOG_NAME** String (may be `null`) => The name of the catalog to which this hierarchy belongs.
- **SCHEMA_NAME** String (may be `null`) => Not supported
- **CUBE_NAME** String => The name of the cube to which this hierarchy belongs.
- **DIMENSION_UNIQUE_NAME** String => The unique name of the dimension to which this hierarchy belongs.
- **HIERARCHY_NAME** String => The name of the hierarchy. Blank if there is only a single hierarchy in the dimension.
- **HIERARCHY_UNIQUE_NAME** String => The unique name of the hierarchy.
- **HIERARCHY_GUID** String (may be `null`) => Hierarchy GUID.
- **HIERARCHY_CAPTION** String => A label or a caption associated with the hierarchy.
- **DIMENSION_TYPE** Short => The type of the dimension.
- **HIERARCHY_CARDINALITY** int => The number of members in the hierarchy.
- **DEFAULT_MEMBER** String (may be `null`) => The default member for this hierarchy.
- **ALL_MEMBER** String (may be `null`) => The member at the highest level of rollup in the hierarchy.
- **DESCRIPTION** String (may be `null`) => A human-readable description of the hierarchy. NULL if no description exists.
- **STRUCTURE** Short => The structure of the hierarchy.
- **IS_VIRTUAL** boolean => Always returns False.
- **IS_READWRITE** boolean => A Boolean that indicates whether the Write Back to dimension column is enabled.
- **DIMENSION_UNIQUE_SETTINGS** int => Always returns MDDIMENSIONS_MEMBER_KEY_UNIQUE (1).
- **DIMENSION_IS_VISIBLE** boolean => Always returns true.
- **HIERARCHY_ORDINAL** int => The ordinal number of the hierarchy across all hierarchies of the cube.
- **DIMENSION_IS_SHARED** boolean => Always returns true.
- **PARENT_CHILD** boolean (may be `null`) => Is hierarchy a parent.

Parameters:

`catalog` - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; `null` means that the catalog name should not be used to narrow the search
`schemaPattern` - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; `null` means that the schema name should not be used to narrow the search
`cubeNamePattern` - a cube name pattern; must match the cube name as it is stored in the database; "" retrieves those without a cube; `null` means that the cube name should not be used to narrow the search
`dimensionUniqueName` - unique name of a dimension (not a pattern); must match the dimension name as it is stored in the database; `null` means that the dimension name should not be used to narrow the search
`hierarchyNamePattern` - a hierarchy name pattern; must match the hierarchy name as it is stored in the database; `null` means that the hierarchy name should not be used to narrow the search

Returns:

a `ResultSet` object in which each row is a hierarchy description

Throws:

`OlapException` - if a database access error occurs

See Also:

`DatabaseMetaData.getSearchStringEscape()`
`Hierarchy`

getLevels

```
public java.sql.ResultSet getLevels(java.lang.String catalog,
         java.lang.String schemaPattern,
         java.lang.String cubeNamePattern,
         java.lang.String dimensionUniqueName,
         java.lang.String hierarchyUniqueName,
         java.lang.String levelNamePattern)
  throws OlapException
```

(continued from last page)

Retrieves a result set describing the Levels in this database.

Specification as for XML/A MDSCHEMA_LEVELS schema rowset.

Each level description has the following columns:

1. **CATALOG_NAME** String (may be `null`) => The name of the catalog to which this level belongs.
2. **SCHEMA_NAME** String (may be `null`) => The name of the schema to which this level belongs.
3. **CUBE_NAME** String => The name of the cube to which this level belongs.
4. **DIMENSION_UNIQUE_NAME** String => The unique name of the dimension to which this level belongs.
5. **HIERARCHY_UNIQUE_NAME** String => The unique name of the hierarchy.
6. **LEVEL_NAME** String => The name of the level.
7. **LEVEL_UNIQUE_NAME** String => The properly escaped unique name of the level.
8. **LEVEL_GUID** String (may be `null`) => Level GUID.
9. **LEVEL_CAPTION** String => A label or caption associated with the hierarchy.
10. **LEVEL_NUMBER** int => The distance of the level from the root of the hierarchy. Root level is zero (0).
11. **LEVEL_CARDINALITY** int => The number of members in the level. This value can be an approximation of the real cardinality.
12. **LEVEL_TYPE** int => Type of the level
13. **CUSTOM_ROLLUP_SETTINGS** int => A bitmap that specifies the custom rollup options.
14. **LEVEL_UNIQUE_SETTINGS** int => A bitmap that specifies which columns contain unique values, if the level only has members with unique names or keys.
15. **LEVEL_IS_VISIBLE** boolean => A Boolean that indicates whether the level is visible.
16. **DESCRIPTION** String (may be `null`) => A human-readable description of the level. NULL if no description exists.

Parameters:

`catalog` - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; `null` means that the catalog name should not be used to narrow the search
`schemaPattern` - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; `null` means that the schema name should not be used to narrow the search
`cubeNamePattern` - a cube name pattern; must match the cube name as it is stored in the database; "" retrieves those without a cube; `null` means that the cube name should not be used to narrow the search
`dimensionUniqueName` - unique name of a dimension (not a pattern); must match the dimension name as it is stored in the database; `null` means that the dimension name should not be used to narrow the search
`hierarchyUniqueName` - unique name of a hierarchy (not a pattern); must match the hierarchy name as it is stored in the database; `null` means that the hierarchy name should not be used to narrow the search
`levelNamePattern` - a level name pattern; must match the level name as it is stored in the database; `null` means that the level name should not be used to narrow the search

Returns:

a `ResultSet` object in which each row is a level description

Throws:

`OlapException` - if a database access error occurs

See Also:

`DatabaseMetaData.getSearchStringEscape()`
`Level`

getMeasures

```
public java.sql.ResultSet getMeasures(java.lang.String catalog,
         java.lang.String schemaPattern,
         java.lang.String cubeNamePattern,
         java.lang.String measureNamePattern,
         java.lang.String measureUniqueName)
  throws OlapException
```

(continued from last page)

Retrieves a result set describing the Measures in this database.

Specification as for XML/A MDSCHEMA_MEASURES schema rowset.

Each measure description has the following columns:

1. **CATALOG_NAME** String (may be `null`) => The name of the catalog to which this measure belongs.
2. **SCHEMA_NAME** String (may be `null`) => The name of the schema to which this measure belongs.
3. **CUBE_NAME** String => The name of the cube to which this measure belongs.
4. **MEASURE_NAME** String => The name of the measure.
5. **MEASURE_UNIQUE_NAME** String => The Unique name of the measure.
6. **MEASURE_CAPTION** String => A label or caption associated with the measure.
7. **MEASURE_GUID** String (may be `null`) => Measure GUID.
8. **MEASURE_AGGREGATOR** int => How a measure was derived.
9. **DATA_TYPE** UnsignedShort => Data type of the measure.
10. **MEASURE_IS_VISIBLE** boolean => A Boolean that always returns True. If the measure is not visible, it will not be included in the schema rowset.
11. **LEVELS_LIST** String (may be `null`) => A string that always returns NULL. EXCEPT that SQL Server returns non-null values!!!
12. **DESCRIPTION** String (may be `null`) => A human-readable description of the measure.

Parameters:

`catalog` - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; `null` means that the catalog name should not be used to narrow the search
`schemaPattern` - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; `null` means that the schema name should not be used to narrow the search
`cubeNamePattern` - a cube name pattern; must match the cube name as it is stored in the database; "" retrieves those without a cube; `null` means that the cube name should not be used to narrow the search
`measureNamePattern` - a measure name pattern; must match the measure name as it is stored in the database; `null` means that the measure name should not be used to narrow the search
`measureUniqueName` - unique name of measure (not a pattern); `null` means that the measure unique name should not be used to narrow the search

Returns:

a `ResultSet` object in which each row is a measure description

Throws:

`OlapException` - if a database access error occurs

See Also:

`DatabaseMetaData.getSearchStringEscape()`
`Measure`

getMembers

```
public java.sql.ResultSet getMembers(java.lang.String catalog,
         java.lang.String schemaPattern,
         java.lang.String cubeNamePattern,
         java.lang.String dimensionUniqueName,
         java.lang.String hierarchyUniqueName,
         java.lang.String levelUniqueName,
         java.lang.String memberUniqueName,
         java.util.Set treeOps)
  throws OlapException
```

(continued from last page)

Retrieves a result set describing the Members in this database.

Specification as for XML/A MDSCHEMA_MEMBERS schema rowset. Rows are sorted by level number then by ordinal.

The `treeOps` parameter allows you to retrieve members relative to a given member. It is only applicable if a `memberUniqueName` is also specified; otherwise it is ignored. The following example retrieves all descendants and ancestors of California, but not California itself:

```
OlapDatabaseMetaData metaData;
ResultSet rset = metaData.getMembers(
    "LOCALDB", "FoodMart", "Sales", null, null, null,
    "[Customers].[USA].[CA]",
    EnumSet.of(Member.TreeOp.ANCESTORS, Member.TreeOp.DESCENDANTS));
```

Each member description has the following columns:

1. **CATALOG_NAME** String (may be `null`) => The name of the catalog to which this member belongs.
2. **SCHEMA_NAME** String (may be `null`) => The name of the schema to which this member belongs.
3. **CUBE_NAME** String => Name of the cube to which this member belongs.
4. **DIMENSION_UNIQUE_NAME** String => Unique name of the dimension to which this member belongs.
5. **HIERARCHY_UNIQUE_NAME** String => Unique name of the hierarchy. If the member belongs to more than one hierarchy, there is one row for each hierarchy to which it belongs.
6. **LEVEL_UNIQUE_NAME** String => Unique name of the level to which the member belongs.
7. **LEVEL_NUMBER** int => The distance of the member from the root of the hierarchy.
8. **MEMBER_ORDINAL** int => Ordinal number of the member. Sort rank of the member when members of this dimension are sorted in their natural sort order. If providers do not have the concept of natural ordering, this should be the rank when sorted by MEMBER_NAME.
9. **MEMBER_NAME** String => Name of the member.
10. **MEMBER_UNIQUE_NAME** String => Unique name of the member.
11. **MEMBER_TYPE** int => Type of the member.
12. **MEMBER_GUID** String (may be `null`) => Memeber GUID.
13. **MEMBER_CAPTION** String => A label or caption associated with the member.
14. **CHILDREN_CARDINALITY** int => Number of children that the member has.
15. **PARENT_LEVEL** int => The distance of the member's parent from the root level of the hierarchy.
16. **PARENT_UNIQUE_NAME** String (may be `null`) => Unique name of the member's parent.
17. **PARENT_COUNT** int => Number of parents that this member has.
18. **TREE_OP** Enumeration (may be `null`) => Tree Operation
19. **DEPTH** int (may be `null`) => depth

Parameters:

`catalog` - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; `null` means that the catalog name should not be used to narrow the search

`schemaPattern` - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; `null` means that the schema name should not be used to narrow the search

`cubeNamePattern` - a cube name pattern; must match the cube name as it is stored in the database; "" retrieves those without a cube; `null` means that the cube name should not be used to narrow the search

`dimensionUniqueName` - unique name of dimension (not a pattern); must match the dimension name as it is stored in the database; `null` means that the dimension name should not be used to narrow the search

`hierarchyUniqueName` - unique name of hierarchy (not a pattern); must match the hierarchy name as it is stored in the database; `null` means that the hierarchy name should not be used to narrow the search

`levelUniqueName` - unique name of level (not a pattern); must match the level name as it is stored in the database; `null` means that the level name should not be used to narrow the search

`memberUniqueName` - unique name of member (not a pattern); `null` means that the measure unique name should not be used to narrow the search

`treeOps` - set of tree operations to retrieve members relative to the member whose unique name was specified; or null to return just the member itself. Ignored if `memberUniqueName` is not specified.

(continued from last page)

Returns:

a `ResultSet` object in which each row is a member description

Throws:

`OlapException` - if a database access error occurs

See Also:

`DatabaseMetaData.getSearchStringEscape()`
`Member`

getSets

```
public java.sql.ResultSet getSets(java.lang.String catalog,
         java.lang.String schemaPattern,
         java.lang.String cubeNamePattern,
         java.lang.String setNamePattern)
  throws OlapException
```

Retrieves a result set describing the named Sets in this database.

Specification as for XML/A MDSCHEMA_SETS schema rowset.

Each set description has the following columns:

1. **CATALOG_NAME** String (may be `null`) => null
2. **SCHEMA_NAME** String (may be `null`) => null
3. **CUBE_NAME** String => null
4. **SET_NAME** String => null
5. **SCOPE** int => null

Parameters:

`catalog` - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; `null` means that the catalog name should not be used to narrow the search
`schemaPattern` - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; `null` means that the schema name should not be used to narrow the search
`cubeNamePattern` - a cube name pattern; must match the cube name as it is stored in the database; "" retrieves those without a cube; `null` means that the cube name should not be used to narrow the search
`setNamePattern` - pattern for the unique name of a set; must match the set name as it is stored in the database; `null` means that the set name should not be used to narrow the search

Returns:

a `ResultSet` object in which each row is a description of a named set

Throws:

`OlapException` - if a database access error occurs

See Also:

`DatabaseMetaData.getSearchStringEscape()`
`NamedSet`

org.olap4j
Interface OlapDataSource

public interface **OlapDataSource**
extends javax.sql.DataSource

A factory for connections to the physical OLAP data source that this `OlapDataSource` object represents.

`OlapDataSource` is a refinement of `javax.sql.DataSource` whose `getConnection` methods return `OlapConnection` objects rather than mere `java.sql.Connection`s.

Method Summary

`OlapConnection`	`getConnection()`
`OlapConnection`	`getConnection(java.lang.String username, java.lang.String password)`

Methods inherited from interface `javax.sql.DataSource`
`getConnection, getConnection, getLoginTimeout, getLogWriter, setLoginTimeout, setLogWriter`

Methods

getConnection

```
public OlapConnection getConnection()
    throws java.sql.SQLException
```

getConnection

```
public OlapConnection getConnection(java.lang.String username,
                java.lang.String password)
    throws java.sql.SQLException
```

org.olap4j
Class OlapException

```
java.lang.Object
  |
  +-java.lang.Throwable
      |
      +-java.lang.Exception
          |
          +-java.sql.SQLException
              |
              +-org.olap4j.OlapException
```

All Implemented Interfaces:
java.io.Serializable

---

public class **OlapException**
extends java.sql.SQLException

An exception describing an error accessing an OLAP database.

Since olap4j extends JDBC, it is natural that `OlapException` should extend JDBC's `java.sql.SQLException`. The implementation by an olap4j driver of a JDBC method which is declared to throw a SQLException may, if the driver chooses, throw instead an OlapException.

OlapException provides some additional information to help an OLAP client identify the location of the error. This information is

---

Nested Class Summary

class	`OlapException.Region` OlapException.Region

Constructor Summary

public	`OlapException(java.lang.String reason, java.lang.String sqlState, int vendorCode)` Constructs a fully-specified `SQLException` object.
public	`OlapException(java.lang.String reason, java.lang.String sqlState)` Constructs an `SQLException` object with the given reason and SQLState; the `vendorCode` field defaults to 0.
public	`OlapException(java.lang.String reason)` Constructs an `SQLException` object with a reason; the `sqlState` field defaults to `null`, and the `vendorCode` field defaults to 0.
public	`OlapException()` Constructs an `SQLException` object; the `reason` field defaults to null, the `sqlState` field defaults to `null`, and the `vendorCode` field defaults to 0.
public	`OlapException(java.lang.String reason, java.lang.Throwable cause)` Constructs an `OlapException` object with a given `reason` and `cause`.

Method Summary

`java.lang.Object`	`getContext()` Returns the context where the exception occurred.
`OlapException.Region`	`getRegion()` Returns the textual region where the exception occurred, or null if no region can be identified.
`void`	`setContext(java.lang.Object context)` Sets the context where the exception occurred.
`void`	`setRegion(OlapException.Region region)` Sets the textual region where the exception occurred.

Methods inherited from class `java.sql.SQLException`
`getErrorCode, getNextException, getSQLState, setNextException`

Methods inherited from class `java.lang.Throwable`
`fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructors

OlapException

```
public OlapException(java.lang.String reason,
                     java.lang.String sqlState,
                     int vendorCode)
```

Constructs a fully-specified `SQLException` object.

Parameters:

`reason` - a description of the exception
`sqlState` - an XOPEN or SQL 99 code identifying the exception
`vendorCode` - a database vendor-specific exception code

OlapException

```
public OlapException(java.lang.String reason,
                     java.lang.String sqlState)
```

Constructs an `SQLException` object with the given reason and SQLState; the `vendorCode` field defaults to 0.

Parameters:

`reason` - a description of the exception
`sqlState` - an XOPEN or SQL 99 code identifying the exception

OlapException

```
public OlapException(java.lang.String reason)
```

(continued from last page)

Constructs an `SQLException` object with a reason; the `sqlState` field defaults to `null`, and the `vendorCode` field defaults to 0.

Parameters:

`reason` - a description of the exception

OlapException

```
public OlapException()
```

Constructs an `SQLException` object; the `reason` field defaults to null, the `sqlState` field defaults to `null`, and the `vendorCode` field defaults to 0.

OlapException

```
public OlapException(java.lang.String reason,
                     java.lang.Throwable cause)
```

Constructs an `OlapException` object with a given `reason` and `cause`.

Parameters:

`reason` - the detail message (which is saved for later retrieval by the `Throwable.getMessage()` method).

`cause` - the cause (which is saved for later retrieval by the `Throwable.getCause()` method). (A null value is permitted, and indicates that the cause is nonexistent or unknown.)

Methods

setRegion

```
public void setRegion(OlapException.Region region)
```

Sets the textual region where the exception occurred.

Parameters:

`region` - Textual region

getRegion

```
public OlapException.Region getRegion()
```

Returns the textual region where the exception occurred, or null if no region can be identified.

Returns:

Region where the exception occurred

setContext

```
public void setContext(java.lang.Object context)
```

Sets the context where the exception occurred.

Parameters:

`context` - Context where the exception occurred

Throws:

`IllegalArgumentException` - If context is not a `Cell` or a `Position`

(continued from last page)

getContext

```
public java.lang.Object getContext()
```

Returns the context where the exception occurred. Typically a `Cell` or a `Position`, or null.

Returns:

context where the exception occurred, or null

org.olap4j
Class OlapException.Region

```
java.lang.Object
  |
  +-org.olap4j.OlapException.Region
```

public static final class **OlapException.Region**
extends java.lang.Object

Description of the position of a syntax or validation error in the source MDX string.

Row and column positions are 1-based and inclusive. For example, in

```
SELECT { [Measures].MEMBERS } ON COLUMNS,
   { } ON ROWS
FROM [Sales]
```

the `SELECT` keyword occupies positions (1, 1) through (1, 6), and would have a `Region(startLine=1, startColumn=1, endColumn=1, endLine=6)`.

Field Summary

public final	endColumn
public final	endLine
public final	startColumn
public final	startLine

Method Summary

java.lang.String	toString()

Methods inherited from class `java.lang.Object`

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Fields

startLine

`public final int startLine`

(continued from last page)

startColumn

```
public final int startColumn
```

endLine

```
public final int endLine
```

endColumn

```
public final int endColumn
```

Methods

toString

```
public java.lang.String toString()
```

org.olap4j
Class OlapExceptionHelper

```
java.lang.Object
  |
  +-org.olap4j.OlapExceptionHelper
```

public class **OlapExceptionHelper**
extends java.lang.Object

Sugar class to help create OlapExceptions.

Constructor Summary

public	OlapExceptionHelper()

Method Summary

static OlapException	createException(Cell context, java.lang.String msg)
static OlapException	createException(Cell context, java.lang.String msg, java.lang.Throwable cause)
static OlapException	createException(java.lang.String msg)
static OlapException	createException(java.lang.String msg, java.lang.Throwable cause)
static OlapException	createException(java.lang.Throwable cause)
static OlapException	toOlapException(java.sql.SQLException e)

Methods inherited from class `java.lang.Object`

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

OlapExceptionHelper

public **OlapExceptionHelper**()

Methods

createException

public static OlapException **createException**(java.lang.String msg)

(continued from last page)

createException

```
public static OlapException createException(java.lang.Throwable cause)
```

createException

```
public static OlapException createException(java.lang.String msg,
         java.lang.Throwable cause)
```

createException

```
public static OlapException createException(Cell context,
         java.lang.String msg)
```

createException

```
public static OlapException createException(Cell context,
         java.lang.String msg,
         java.lang.Throwable cause)
```

toOlapException

```
public static OlapException toOlapException(java.sql.SQLException e)
```

org.olap4j
Interface OlapParameterMetaData

public interface **OlapParameterMetaData**
extends java.sql.ParameterMetaData

Extension to `java.sql.ParameterMetaData` for parameters of OLAP statements.

Chief differences:

- An OLAP statement parameter has a name.
- An OLAP statement parameter may be a member. If this is the case, the `ParameterMetaData.getParameterType(int)` method returns `Types.OTHER`.
- An additional method `getParameterOlapType(int)` provides extra type information; in particular, the hierarchy that a member parameter belongs to.

Parameters to an OLAP statement must have default values, and therefore it is not necessary to set every parameter.

Fields inherited from interface `java.sql.ParameterMetaData`
`parameterModeIn`, `parameterModeInOut`, `parameterModeOut`, `parameterModeUnknown`, `parameterNoNulls`, `parameterNullable`, `parameterNullableUnknown`

Method Summary

`java.lang.String`	`getParameterName(int param)` Returns the name of this parameter.
`Type`	`getParameterOlapType(int param)` Retrieves the designated parameter's OLAP type.

Methods inherited from interface `java.sql.ParameterMetaData`
`getParameterClassName, getParameterCount, getParameterMode, getParameterType, getParameterTypeName, getPrecision, getScale, isNullable, isSigned`

Methods

getParameterName

```
public java.lang.String getParameterName(int param)
  throws OlapException
```

Returns the name of this parameter.

Parameters:
`param` - the first parameter is 1, the second is 2, ...

Returns:
parameter name

Throws:
`OlapException` - if a database access error occurs

getParameterOlapType

```
public Type getParameterOlapType(int param)
  throws OlapException
```

Retrieves the designated parameter's OLAP type.

Parameters:

`param` - the first parameter is 1, the second is 2, ...

Returns:

OLAP type

Throws:

`OlapException` - if a database access error occurs

org.olap4j
Interface OlapStatement

All Superinterfaces:

OlapWrapper

All Subinterfaces:

PreparedOlapStatement

public interface **OlapStatement**
extends java.sql.Statement, OlapWrapper

Object used for statically executing an MDX statement and returning a `CellSet`.

An `OlapStatement` is generally created using `createStatement()`.

See Also:

`PreparedOlapStatement`

Fields inherited from interface `java.sql.Statement`
`CLOSE_ALL_RESULTS`, `CLOSE_CURRENT_RESULT`, `EXECUTE_FAILED`, `KEEP_CURRENT_RESULT`, `NO_GENERATED_KEYS`, `RETURN_GENERATED_KEYS`, `SUCCESS_NO_INFO`

Method Summary

`CellSet`	`executeOlapQuery(SelectNode selectNode)` Executes an OLAP statement expressed as a parse tree.
`CellSet`	`executeOlapQuery(java.lang.String mdx)` Executes an OLAP statement.

Methods inherited from interface `java.sql.Statement`
`addBatch, cancel, clearBatch, clearWarnings, close, execute, execute, execute, execute, executeBatch, executeQuery, executeUpdate, executeUpdate, executeUpdate, executeUpdate, getConnection, getFetchDirection, getFetchSize, getGeneratedKeys, getMaxFieldSize, getMaxRows, getMoreResults, getMoreResults, getQueryTimeout, getResultSet, getResultSetConcurrency, getResultSetHoldability, getResultSetType, getUpdateCount, getWarnings, setCursorName, setEscapeProcessing, setFetchDirection, setFetchSize, setMaxFieldSize, setMaxRows, setQueryTimeout`

Methods inherited from interface `org.olap4j.OlapWrapper`
`isWrapperFor, unwrap`

Methods

executeOlapQuery

```
public CellSet executeOlapQuery(java.lang.String mdx)
  throws OlapException
```

Executes an OLAP statement.

(continued from last page)

Parameters:

`mdx` - MDX `SELECT` statement

Returns:

Cell set

Throws:

`OlapException` - if a database access error occurs, this method is called on a closed `OlapStatement`, the query times out (see `Statement.setQueryTimeout(int)`) or another thread cancels the statement (see `Statement.cancel()`)

executeOlapQuery

```
public CellSet executeOlapQuery(SelectNode selectNode)
  throws OlapException
```

Executes an OLAP statement expressed as a parse tree.

Validates the parse tree before executing it.

Parameters:

`selectNode` - Parse tree of MDX `SELECT` statement

Returns:

Cell set

Throws:

`OlapException` - if a database access error occurs, this method is called on a closed `OlapStatement`, the query times out (see `Statement.setQueryTimeout(int)`) or another thread cancels the statement (see `Statement.cancel()`)

org.olap4j
Interface OlapWrapper

All Subinterfaces:

CellSet, CellSetMetaData, OlapConnection, OlapDatabaseMetaData, OlapStatement, PreparedOlapStatement

public interface **OlapWrapper**
extends

Interface for olap4j classes which provide the ability to retrieve the delegate instance when the instance in question is in fact a proxy class.

`OlapWrapper` duplicates the functionality of the `java.sql.Wrapper` interface (introduced in JDBC 4.0), making this functionality available to olap4j clients running in a JDBC 3.0 environment. For code which will run only on JDBC 4.0 and later, Wrapper can be used, and OlapWrapper can be ignored.

In JDBC 3.0 (JDK 1.5) and earlier, the `OlapWrapper` interface is used to convert a JDBC class to the corresponding olap4j class. For instance, write

```
import java.sql.Connection;
import java.sql.DriverManager;
import org.olap4j.OlapConnection;
import org.olap4j.OlapWrapper;

Connection connection = DriverManager.getConnection("jdbc: ...");
OlapWrapper wrapper = (OlapWrapper) connection;
OlapConnection olapConnection = wrapper.unwrap(OlapConnection.class);
```

to create a JDBC 3.0 connection and convert it to an olap4j connection.

In JDBC 4.0 (JDK 1.6) and later, you don't need to use this class. All of the key JDBC classes implement `java.sql.Wrapper` interface, so you can use its `isWrapper` and `unwrap` methods without casting. For instance, write

```
import java.sql.Connection;
import java.sql.DriverManager;
import org.olap4j.OlapConnection;

Connection connection = DriverManager.getConnection("jdbc: ...");
OlapConnection olapConnection = connection.unwrap(OlapConnection.class);
```

to create a JDBC 4.0 connection and convert it to an olap4j connection.

Method Summary

`boolean`	`isWrapperFor(java.lang.Class iface)`
`java.lang.Object`	`unwrap(java.lang.Class iface)`

Methods

(continued from last page)

unwrap

```
public java.lang.Object unwrap(java.lang.Class iface)
  throws java.sql.SQLException
```

isWrapperFor

```
public boolean isWrapperFor(java.lang.Class iface)
  throws java.sql.SQLException
```

org.olap4j
Interface Position

public interface **Position**
extends

Position on one of the `CellSetAxis` objects in a `CellSet`.

An axis has a particular dimensionality, that is, a set of one or more dimensions which will appear on that axis, and every position on that axis will have a member of each of those dimensions. For example, in the MDX query`SELECT {[Measures].[Unit Sales], [Measures].[Store Sales]} ON COLUMNS,`

```
     CrossJoin(
         {[Gender].Members},
         {[Product].[Food], [Product].[Drink]}) ON ROWS
 FROM [Sales]
```

the `COLUMNS` axis has dimensionality `{[Measures]}` and the `ROWS` axis has dimensionality `{[Gender], [Product]}`. In the result,

Gender	***Product***	**Unit Sales**	**Store Sales**
All Gender	**Food**	191,940	409,035.59
All Gender	**Drink**	24,597	48,836.21
F	**Food**	94,814	203,094.17
F	**Drink**	12,202	24,457.37
M	**Food**	97,126	205,941.42
M	**Drink**	12,395	24,378.84

each of the six positions on the `ROWS` axis has two members, consistent with its dimensionality of 2. The `COLUMNS` axis has two positions, each with one member.

Method Summary

`java.util.List`	`getMembers()` Returns the list of Member objects at this position.
`int`	`getOrdinal()` Returns the zero-based ordinal of this Position on its `CellSetAxis`.

Methods

getMembers

`public java.util.List` **`getMembers`**`()`

Returns the list of Member objects at this position.

Recall that the `getHierarchies()` method describes the hierarchies which occur on an axis. The positions on that axis must conform. Suppose that the ROWS axis of a given statement returns `{[Gender], [Store]}`. Then every Position on that axis will have two members: the first a member of the [Gender] dimension, the second a member of the [Store] dimension.

Returns:

A list of Member objects at this Position.

getOrdinal

```
public int getOrdinal()
```

Returns the zero-based ordinal of this Position on its `CellSetAxis`.

Returns:
ordinal of this Position

org.olap4j
Interface PreparedOlapStatement

All Superinterfaces:
OlapStatement, OlapWrapper

---

public interface **PreparedOlapStatement**
extends java.sql.PreparedStatement, OlapStatement

An object that represents a precompiled OLAP statement.

An OLAP statement is precompiled and stored in a `PreparedOlapStatement` object. This object can then be used to efficiently execute this statement multiple times.

A `PreparedOlapStatement` is generally created using `prepareOlapStatement(String)`.

Note: The setter methods (`setShort`, `setString`, and so on) for setting IN parameter values must specify types that are compatible with the defined type of the input parameter. For instance, if the IN parameter has type `INTEGER`, then the method `setInt` should be used.

If a parameter has Member type, use the `PreparedStatement.setObject(int, java.lang.Object)` method to set it. A `OlapException` will be thrown if the object is not an instance of `Member` or does not belong to the correct `Hierarchy`.

The method `getParameterMetaData()` returns a description of the parameters, as in JDBC. The result is an `OlapParameterMetaData`.

Unlike JDBC, it is not necessary to assign a value to every parameter. This is because OLAP parameters have a default value. Parameters have their default value until they are set, and then retain their new values for each subsequent execution of this `PreparedOlapStatement`.

See Also:
`prepareOlapStatement(String)`, `CellSet`

Fields inherited from interface `java.sql.Statement`

`CLOSE_ALL_RESULTS`, `CLOSE_CURRENT_RESULT`, `EXECUTE_FAILED`, `KEEP_CURRENT_RESULT`, `NO_GENERATED_KEYS`, `RETURN_GENERATED_KEYS`, `SUCCESS_NO_INFO`

Fields inherited from interface `java.sql.Statement`

`CLOSE_ALL_RESULTS`, `CLOSE_CURRENT_RESULT`, `EXECUTE_FAILED`, `KEEP_CURRENT_RESULT`, `NO_GENERATED_KEYS`, `RETURN_GENERATED_KEYS`, `SUCCESS_NO_INFO`

Method Summary

`CellSet`	`executeQuery()` Executes the MDX query in this `PreparedOlapStatement` object and returns the `CellSet` object generated by the query.
`Cube`	`getCube()` Returns the cube (or virtual cube) which this statement is based upon.
`CellSetMetaData`	`getMetaData()` Retrieves a `CellSetMetaData` object that contains information about the axes and cells of the `CellSet` object that will be returned when this `PreparedOlapStatement` object is executed.

`OlapParameterMetaData`	`getParameterMetaData()` Retrieves the number, types and properties of this `PreparedOlapStatement` object's parameters.

Methods inherited from interface `java.sql.PreparedStatement`
`addBatch, clearParameters, execute, executeQuery, executeUpdate, getMetaData, getParameterMetaData, setArray, setAsciiStream, setBigDecimal, setBinaryStream, setBlob, setBoolean, setByte, setBytes, setCharacterStream, setClob, setDate, setDate, setDouble, setFloat, setInt, setLong, setNull, setNull, setObject, setObject, setObject, setRef, setShort, setString, setTime, setTime, setTimestamp, setTimestamp, setUnicodeStream, setURL`

Methods inherited from interface `java.sql.Statement`
`addBatch, cancel, clearBatch, clearWarnings, close, execute, execute, execute, execute, executeBatch, executeQuery, executeUpdate, executeUpdate, executeUpdate, executeUpdate, getConnection, getFetchDirection, getFetchSize, getGeneratedKeys, getMaxFieldSize, getMaxRows, getMoreResults, getMoreResults, getQueryTimeout, getResultSet, getResultSetConcurrency, getResultSetHoldability, getResultSetType, getUpdateCount, getWarnings, setCursorName, setEscapeProcessing, setFetchDirection, setFetchSize, setMaxFieldSize, setMaxRows, setQueryTimeout`

Methods inherited from interface `org.olap4j.OlapStatement`
`executeOlapQuery, executeOlapQuery`

Methods inherited from interface `java.sql.Statement`
`addBatch, cancel, clearBatch, clearWarnings, close, execute, execute, execute, execute, executeBatch, executeQuery, executeUpdate, executeUpdate, executeUpdate, executeUpdate, getConnection, getFetchDirection, getFetchSize, getGeneratedKeys, getMaxFieldSize, getMaxRows, getMoreResults, getMoreResults, getQueryTimeout, getResultSet, getResultSetConcurrency, getResultSetHoldability, getResultSetType, getUpdateCount, getWarnings, setCursorName, setEscapeProcessing, setFetchDirection, setFetchSize, setMaxFieldSize, setMaxRows, setQueryTimeout`

Methods inherited from interface `org.olap4j.OlapWrapper`
`isWrapperFor, unwrap`

Methods

executeQuery

```
public CellSet executeQuery()
  throws OlapException
```

Executes the MDX query in this `PreparedOlapStatement` object and returns the `CellSet` object generated by the query.

Returns:

an `CellSet` object that contains the data produced by the query; never `null`

Throws:

`OlapException` - if a database access error occurs

getParameterMetaData

```
public OlapParameterMetaData getParameterMetaData()
  throws OlapException
```

Retrieves the number, types and properties of this `PreparedOlapStatement` object's parameters.

Returns:

an `OlapParameterMetaData` object that contains information about the number, types and properties of this `PreparedOlapStatement` object's parameters

Throws:

`OlapException` - if a database access error occurs

See Also:

`OlapParameterMetaData`

getMetaData

```
public CellSetMetaData getMetaData()
  throws java.sql.SQLException
```

Retrieves a `CellSetMetaData` object that contains information about the axes and cells of the `CellSet` object that will be returned when this `PreparedOlapStatement` object is executed.

Returns:

the description of this `CellSet`'s axes and cells

Throws:

`OlapException` - if a database access error occurs

getCube

```
public Cube getCube()
```

Returns the cube (or virtual cube) which this statement is based upon.

Returns:

cube this statement is based upon

Package org.olap4j.driver.xmla

olap4j to XML for Analysis bridge.

org.olap4j.driver.xmla
Class XmlaOlap4jDriver

```
java.lang.Object
  |
  +-org.olap4j.driver.xmla.XmlaOlap4jDriver
```

All Implemented Interfaces:
java.sql.Driver

---

public class **XmlaOlap4jDriver**
extends java.lang.Object
implements java.sql.Driver

Olap4j driver for generic XML for Analysis (XMLA) providers.

Since olap4j is a superset of JDBC, you register this driver as you would any JDBC driver:
`Class.forName("org.olap4j.driver.xmla.XmlaOlap4jDriver");`Then create a connection using a URL with the prefix "jdbc:xmla:". For example, `import java.sql.Connection;`

```
 import java.sql.DriverManager;
 import org.olap4j.OlapConnection;
 Connection connection =
    DriverManager.getConnection(
       "jdbc:xmla:");
 OlapConnection olapConnection =
    connection.unwrap(OlapConnection.class);
```

Note how we use the `unwrap(Class)` method to down-cast the JDBC connection object to the extension `OlapConnection` object. This method is only available in JDBC 4.0 (JDK 1.6 onwards).

Connection properties

Unless otherwise stated, properties are optional. If a property occurs multiple times in the connect string, the first occurrence is used.

Property	Description
Server	URL of HTTP server. Required.
Catalog	Catalog name to use. By default, the first one returned by the XMLA server will be used.
Provider	Name of the XMLA provider.
DataSource	Name of the XMLA datasource. When using a Mondrian backed XMLA server, be sure to include the full datasource name between quotes.
Cache	Class name of the SOAP cache to use. A built in memory cache is available with org.olap4j.driver.xmla.cache.XmlaOlap4jNamedMemoryCache. Has to be an implementation of IXmlaOlap4jCache.By default, no SOAP query cache will be used.
Cache.*	Properties to transfer to the selected cache implementation. See IXmlaOlap4jCache or your selected implementation for properties details.
TestProxyCookie	String that uniquely identifies a proxy object in `PROXY_MAP` via which to send XMLA requests for testing purposes.

Nested Class Summary

class	XmlaOlap4jDriver.Property XmlaOlap4jDriver.Property
class	XmlaOlap4jDriver.Proxy XmlaOlap4jDriver.Proxy

Field Summary

public static final	MAJOR_VALUE Value: **0**
public static final	MINOR_VERSION Value: **905**
public static final	NAME Value: **olap4j driver for XML/A**
public static final	PROXY_MAP For testing.
public static final	VERSION Value: **0.9.5**

Method Summary

boolean	acceptsURL(java.lang.String url)
java.sql.Connection	connect(java.lang.String url, java.util.Properties info)
static java.util.concurrent.Future	getFuture(XmlaOlap4jProxy proxy, java.net.URL url, java.lang.String request) Returns a future object representing an asynchronous submission of an XMLA request to a URL.
int	getMajorVersion()
int	getMinorVersion()
java.sql.DriverPropertyInfo[]	getPropertyInfo(java.lang.String url, java.util.Properties info)
boolean	jdbcCompliant()
static java.lang.String	nextCookie() Generates and returns a unique string.

Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface java.sql.Driver
acceptsURL, connect, getMajorVersion, getMinorVersion, getPropertyInfo, jdbcCompliant

Fields

NAME

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

Constant value: **`olap4j driver for XML/A`**

VERSION

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

Constant value: **`0.9.5`**

MAJOR_VERSION

```
public static final int MAJOR_VERSION
```

Constant value: **`0`**

MINOR_VERSION

```
public static final int MINOR_VERSION
```

Constant value: **`905`**

PROXY_MAP

```
public static final java.util.Map PROXY_MAP
```

For testing. Map from a cookie value (which is uniquely generated for each test) to a proxy object. Uses a weak hash map so that, if the code that created the proxy 'forgets' the cookie value, then the proxy can be garbage-collected.

Methods

connect

```
public java.sql.Connection connect(java.lang.String url,
         java.util.Properties info)
  throws java.sql.SQLException
```

acceptsURL

```
public boolean acceptsURL(java.lang.String url)
  throws java.sql.SQLException
```

(continued from last page)

getPropertyInfo

```
public java.sql.DriverPropertyInfo[] getPropertyInfo(java.lang.String url,
         java.util.Properties info)
  throws java.sql.SQLException
```

getMajorVersion

```
public int getMajorVersion()
```

getMinorVersion

```
public int getMinorVersion()
```

jdbcCompliant

```
public boolean jdbcCompliant()
```

getFuture

```
public static java.util.concurrent.Future getFuture(XmlaOlap4jProxy proxy,
         java.net.URL url,
         java.lang.String request)
```

Returns a future object representing an asynchronous submission of an XMLA request to a URL.

Parameters:

`proxy` - Proxy via which to send the request
`url` - URL of XMLA server
`request` - Request

Returns:

Future object from which the byte array containing the result of the XMLA call can be obtained

nextCookie

```
public static java.lang.String nextCookie()
```

Generates and returns a unique string.

Returns:

unique string

org.olap4j.driver.xmla
Class XmlaOlap4jDriver.Property

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.driver.xmla.XmlaOlap4jDriver.Property
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

public static final class **XmlaOlap4jDriver.Property**
extends java.lang.Enum

Properties supported by this driver.

Field Summary

public static final	Cache
public static final	Catalog
public static final	DataSource
public static final	Provider
public static final	Server
public static final	TestProxyCookie

Method Summary

static XmlaOlap4jDriver.Property	valueOf(java.lang.String name)
static XmlaOlap4jDriver.Property[]	values()

Methods inherited from class `java.lang.Enum`

compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

Methods inherited from class `java.lang.Object`

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface `java.lang.Comparable`

compareTo

Fields

TestProxyCookie

`public static final org.olap4j.driver.xmla.XmlaOlap4jDriver.Property` **`TestProxyCookie`**

Server

`public static final org.olap4j.driver.xmla.XmlaOlap4jDriver.Property` **`Server`**

Catalog

`public static final org.olap4j.driver.xmla.XmlaOlap4jDriver.Property` **`Catalog`**

Provider

`public static final org.olap4j.driver.xmla.XmlaOlap4jDriver.Property` **`Provider`**

DataSource

`public static final org.olap4j.driver.xmla.XmlaOlap4jDriver.Property` **`DataSource`**

Cache

`public static final org.olap4j.driver.xmla.XmlaOlap4jDriver.Property` **`Cache`**

Methods

values

`public final static XmlaOlap4jDriver.Property[]` **`values`**`()`

valueOf

`public static XmlaOlap4jDriver.Property` **`valueOf`**`(java.lang.String name)`

org.olap4j.driver.xmla
Interface XmlaOlap4jDriver.Proxy

All Superinterfaces:
XmlaOlap4jProxy

---

public interface **XmlaOlap4jDriver.Proxy**
extends XmlaOlap4jProxy

This is a mock subclass to prevent retro-compatibility issues. If you're using this class, please change your code to use XmlaOlap4jProxy instead.

Methods inherited from interface `org.olap4j.driver.xmla.proxy.XmlaOlap4jProxy`
`get, getEncodingCharsetName, submit`

Package
org.olap4j.driver.xmla.cache

Provides SOAP caching functionality.

org.olap4j.driver.xmla.cache
Interface XmlaOlap4jCache

All Known Implementing Classes:
XmlaOlap4jNamedMemoryCache

public interface **XmlaOlap4jCache**
extends

XMLA driver cache. Implementations will have to declare those methods.

The XMLA driver will call the cache before each SOAP request to see if it wasn't sent previously and if a SOAP response doesn't already exist in it.

Any implementations have to declare a constructor which takes a String as a parameter. This string value is the unique name of the connection which triggered the request.

Method Summary

`void`	`flushCache()` Tells the cache to flush all cached entries.
`byte[]`	`get(java.lang.String id, java.net.URL url, byte[] request)` Fetches a SOAP response from the cache.
`void`	`put(java.lang.String id, java.net.URL url, byte[] request, byte[] response)` Adds a SOAP response to the cache.
`java.lang.String`	`setParameters(java.util.Map config, java.util.Map props)` Convenience method to receive custom properties.

Methods

get

```
public byte[] get(java.lang.String id,
                  java.net.URL url,
                  byte[] request)
    throws XmlaOlap4jInvalidStateException
```

Fetches a SOAP response from the cache. Returns null if there are no cached response corresponding to the SOAP message and the URL.

Parameters:

`id` - The connection unique name which called this cache.
`url` - The URL where the SOAP message was sent.
`request` - The SOAP complete message.

Returns:

The SOAP response, null if there are no corresponding response in the cache.

Throws:

`XmlaOlap4jInvalidStateException` - when operations to the cache are performed but it hasn't been initialized. Make sure you call the setParameters method.

put

```
public void put(java.lang.String id,
         java.net.URL url,
         byte[] request,
         byte[] response)
  throws XmlaOlap4jInvalidStateException
```

Adds a SOAP response to the cache. It has to be relative to the URL of the SOAP service.

Parameters:

`id` - The connection unique name which called this cache.

`url` - The URL of the SOAP endpoint.

`request` - The full SOAP message from which we want to cache its response.

`response` - The response to cache.

Throws:

`XmlaOlap4jInvalidStateException` - when operations to the cache are performed but it hasn't been initialized. Make sure you call the setParameters method.

flushCache

```
public void flushCache()
```

Tells the cache to flush all cached entries.

setParameters

```
public java.lang.String setParameters(java.util.Map config,
         java.util.Map props)
```

Convenience method to receive custom properties.

The XMLA driver takes cache properties as "`Cache.[property name]=[value]`" in its JDBC url. All those properties should be striped of their "`Cache.`" prefix and sent to this method as the properties parameter.

Also, the complete config map of the current connection should be passed as the config parameter.

Parameters:

`config` - The complete configuration parameters which were used to create the current connection.

`props` - The properties received from the JDBC url.

Returns:

Returns a string object which gives a reference id to the caller for future use. This id has to be passed along with any future get and put requests.

org.olap4j.driver.xmla.cache
Class XmlaOlap4jNamedMemoryCache

```
java.lang.Object
  |
  +-org.olap4j.driver.xmla.cache.XmlaOlap4jNamedMemoryCache
```

All Implemented Interfaces:
XmlaOlap4jCache

public class **XmlaOlap4jNamedMemoryCache**
extends java.lang.Object
implements XmlaOlap4jCache

Implementation of the XMLA SOAP cache that places its cache entries in memory for later use. It is thread safe and at static class level.

It supports cache sharing through the Name property.

All parameters are optional.

- **Name**
  A unique identifier which allows two connections to share a same cache space. Setting this to an already existing cache space will cause the cache manager to ignore other configuration properties, such as eviction mode and so on. Not setting this property will assign a random name to the cache space, thus creating a unique space.
- **Size**
  The number of entries to maintain in cache under the given cache name.
- **Timeout**
  The number of seconds to maintain entries in cache before expiration.
- **Mode**
  Supported eviction modes are LIFO (last in first out), FIFO (first in first out), LFU (least frequently used) and MFU (most frequently used)

See Also:
`XmlaOlap4jNamedMemoryCache$Property`

Nested Class Summary

`class`	`XmlaOlap4jNamedMemoryCache.Mode` XmlaOlap4jNamedMemoryCache.Mode
`class`	`XmlaOlap4jNamedMemoryCache.Property` XmlaOlap4jNamedMemoryCache.Property

Constructor Summary

`public`	`XmlaOlap4jNamedMemoryCache()` Default constructor which instantiates the concurrent hash map.

Method Summary

`void`	`flushCache()`
`byte[]`	`get(java.lang.String id, java.net.URL url, byte[] request)`

void	put(java.lang.String id, java.net.URL url, byte[] request, byte[] response)
java.lang.String	setParameters(java.util.Map config, java.util.Map props)

Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface org.olap4j.driver.xmla.cache.XmlaOlap4jCache
flushCache, get, put, setParameters

Constructors

XmlaOlap4jNamedMemoryCache

```
public XmlaOlap4jNamedMemoryCache()
```

Default constructor which instantiates the concurrent hash map.

Methods

setParameters

```
public java.lang.String setParameters(java.util.Map config,
         java.util.Map props)
```

get

```
public byte[] get(java.lang.String id,
         java.net.URL url,
         byte[] request)
  throws XmlaOlap4jInvalidStateException
```

put

```
public void put(java.lang.String id,
         java.net.URL url,
         byte[] request,
         byte[] response)
  throws XmlaOlap4jInvalidStateException
```

flushCache

```
public void flushCache()
```

org.olap4j.driver.xmla.cache
Class XmlaOlap4jNamedMemoryCache.Property

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.driver.xmla.cache.XmlaOlap4jNamedMemoryCache.Property
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

public static final class **XmlaOlap4jNamedMemoryCache.Property**
extends java.lang.Enum

Properties which will be considered for configuration.

All parameters are optional.

Field Summary

`public static final`	`Mode` Eviction mode.
`public static final`	`Name` A unique identifier which allows two connections to share a same cache space.
`public static final`	`Size` The number of entries to maintain in cache under the given cache name.
`public static final`	`Timeout` The number of seconds to maintain entries in cache before expiration.

Method Summary

`static XmlaOlap4jNamedMemoryCache.Property`	`valueOf(java.lang.String name)`
`static XmlaOlap4jNamedMemoryCache.Property[]`	`values()`

Methods inherited from class `java.lang.Enum`

`compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `java.lang.Comparable`

`compareTo`

(continued from last page)

Fields

Name

```
public static final org.olap4j.driver.xmla.cache.XmlaOlap4jNamedMemoryCache.Property
Name
```

A unique identifier which allows two connections to share a same cache space. Setting this to an already existing cache space will cause the cache manager to ignore other configuration properties, such as eviction mode and so on. Not setting this property will assign a random name to the cache space, thus creating a unique space.

Size

```
public static final org.olap4j.driver.xmla.cache.XmlaOlap4jNamedMemoryCache.Property
Size
```

The number of entries to maintain in cache under the given cache name.

Timeout

```
public static final org.olap4j.driver.xmla.cache.XmlaOlap4jNamedMemoryCache.Property
Timeout
```

The number of seconds to maintain entries in cache before expiration.

Mode

```
public static final org.olap4j.driver.xmla.cache.XmlaOlap4jNamedMemoryCache.Property
Mode
```

Eviction mode. Supported eviction modes are LIFO (last in first out), FIFO (first in first out), LFU (least frequently used) and MFU (most frequently used).

Methods

values

```
public final static XmlaOlap4jNamedMemoryCache.Property[] values()
```

valueOf

```
public static XmlaOlap4jNamedMemoryCache.Property valueOf(java.lang.String name)
```

org.olap4j.driver.xmla.cache
Class XmlaOlap4jNamedMemoryCache.Mode

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.driver.xmla.cache.XmlaOlap4jNamedMemoryCache.Mode
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

public static final class **XmlaOlap4jNamedMemoryCache.Mode**
extends java.lang.Enum

Defines the supported eviction modes.

Field Summary

public static final	FIFO First-in, first-out.
public static final	LFU Least-frequently used.
public static final	LIFO Last-in, first-out.
public static final	MFU Most-frequently used.

Method Summary

static XmlaOlap4jNamedMemoryCache.Mode	valueOf(java.lang.String name)
static XmlaOlap4jNamedMemoryCache.Mode[]	values()

Methods inherited from class java.lang.Enum

compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface java.lang.Comparable

compareTo

Fields

(continued from last page)

LIFO

```
public static final org.olap4j.driver.xmla.cache.XmlaOlap4jNamedMemoryCache.Mode LIFO
```

Last-in, first-out.

FIFO

```
public static final org.olap4j.driver.xmla.cache.XmlaOlap4jNamedMemoryCache.Mode FIFO
```

First-in, first-out.

LFU

```
public static final org.olap4j.driver.xmla.cache.XmlaOlap4jNamedMemoryCache.Mode LFU
```

Least-frequently used.

MFU

```
public static final org.olap4j.driver.xmla.cache.XmlaOlap4jNamedMemoryCache.Mode MFU
```

Most-frequently used.

Methods

values

```
public final static XmlaOlap4jNamedMemoryCache.Mode[] values()
```

valueOf

```
public static XmlaOlap4jNamedMemoryCache.Mode valueOf(java.lang.String name)
```

Package org.olap4j.driver.xmla.proxy

Provides proxy communications with XML/A servers.

org.olap4j.driver.xmla.proxy
Interface XmlaOlap4jCachedProxy

All Superinterfaces:
XmlaOlap4jProxy

---

public interface **XmlaOlap4jCachedProxy**
extends XmlaOlap4jProxy

Extended Proxy interface which supports cached SOAP calls.

Method Summary

`void`	`setCache(java.util.Map configParameters, java.util.Map properties)` Sets the cache class to use as a SOAP message cache.

Methods inherited from interface `org.olap4j.driver.xmla.proxy.XmlaOlap4jProxy`

`get`, `getEncodingCharsetName`, `submit`

Methods

setCache

```
public void setCache(java.util.Map configParameters,
                     java.util.Map properties)
  throws OlapException
```

Sets the cache class to use as a SOAP message cache.

Calling this method is not mandatory. If it isn't called, no cache will be used and all SOAP requests will be sent to the service end-point.

Parameters:

`configParameters` - This contains all the parameters used to configure the Olap4j driver. It contains the full class name of the cache implementation to use as well as the raw Cache config parameters.
`properties` - The properties to configure the cache, so all config parameters which started by Cache.* are inside this convenient thigny.

See Also:

`XmlaOlap4jCache`

org.olap4j.driver.xmla.proxy
Class XmlaOlap4jHttpProxy

```
java.lang.Object
  |
  +-org.olap4j.driver.xmla.proxy.XmlaOlap4jAbstractHttpProxy
      |
      +-org.olap4j.driver.xmla.proxy.XmlaOlap4jHttpProxy
```

All Implemented Interfaces:
XmlaOlap4jCachedProxy

public class **XmlaOlap4jHttpProxy**
extends XmlaOlap4jAbstractHttpProxy

Extends the AbstractCachedProxy and serves as a production ready http communication class. Every SOAP request sends a POST call to the destination XMLA server and returns the response as a byte array, conforming to the Proxy interface.

It also takes advantage of the AbstractHttpProxy cookie managing facilities. All cookies received from the end point server will be sent back if they are not expired and they also conform to cookie domain rules.

Constructor Summary

public	`XmlaOlap4jHttpProxy()`

Method Summary

java.lang.String	`getEncodingCharsetName()`
byte[]	`getResponse(java.net.URL url, java.lang.String request)`
java.util.concurrent. Future	`getResponseViaSubmit(java.net.URL url, java.lang.String request)`

Methods inherited from class `org.olap4j.driver.xmla.proxy.XmlaOlap4jAbstractHttpProxy`

`get, getResponse, getResponseViaSubmit, setCache, submit`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.driver.xmla.proxy.XmlaOlap4jCachedProxy`

`setCache`

Methods inherited from interface `org.olap4j.driver.xmla.proxy.XmlaOlap4jProxy`

`get, getEncodingCharsetName, submit`

Constructors

(continued from last page)

XmlaOlap4jHttpProxy

```
public XmlaOlap4jHttpProxy()
```

Methods

getResponse

```
public byte[] getResponse(java.net.URL url,
         java.lang.String request)
  throws XmlaOlap4jProxyException
```

Sends a request to a URL and returns the response.

getResponseViaSubmit

```
public java.util.concurrent.Future getResponseViaSubmit(java.net.URL url,
         java.lang.String request)
```

Submits a request for background execution.

getEncodingCharsetName

```
public java.lang.String getEncodingCharsetName()
```

org.olap4j.driver.xmla.proxy
Interface XmlaOlap4jProxy

All Subinterfaces:
XmlaOlap4jCachedProxy, Proxy

public interface **XmlaOlap4jProxy**
extends

Defines a common set of methods for proxy objects.

Method Summary

`byte[]`	`get(java.net.URL url, java.lang.String request)` Sends a request to a URL and returns the response.
`java.lang.String`	`getEncodingCharsetName()` Returns the name of the character set use for encoding the XML string.
`java.util.concurrent.Future`	`submit(java.net.URL url, java.lang.String request)` Submits a request for background execution.

Methods

get

```
public byte[] get(java.net.URL url,
          java.lang.String request)
  throws XmlaOlap4jProxyException,
          java.io.IOException
```

Sends a request to a URL and returns the response.

Parameters:
`url` - Target URL
`request` - Request string

Returns:
Response The byte array that contains the whole response from the server.

Throws:
`IOException` - This exception declaration will be removed soon. Don't catch this. Catch XmlaOlap4jProxyException instead.
`XmlaOlap4jProxyException` - If anything occurs during the request execution.

submit

```
public java.util.concurrent.Future submit(java.net.URL url,
          java.lang.String request)
```

Submits a request for background execution.

Parameters:
`url` - URL

(continued from last page)

`request` - Request

Returns:

Future object representing the submitted job

getEncodingCharsetName

```
public java.lang.String getEncodingCharsetName()
```

Returns the name of the character set use for encoding the XML string.

org.olap4j.driver.xmla.proxy
Class XmlaOlap4jProxyException

```
java.lang.Object
  |
  +-java.lang.Throwable
      |
      +-java.lang.Exception
          |
          +-org.olap4j.driver.xmla.proxy.XmlaOlap4jProxyException
```

All Implemented Interfaces:
java.io.Serializable

public class **XmlaOlap4jProxyException**
extends java.lang.Exception

Gets thrown whenever an exception is encountered during the querying of an XmlaOlap4jProxy subclass.

Constructor Summary

public	XmlaOlap4jProxyException(java.lang.String message, java.lang.Throwable cause)

Methods inherited from class `java.lang.Throwable`

fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString

Methods inherited from class `java.lang.Object`

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

XmlaOlap4jProxyException

```
public XmlaOlap4jProxyException(java.lang.String message,
                                java.lang.Throwable cause)
```

Package
org.olap4j.mdx

Provides an object model to represent statements and expressions in the MDX lanaugage as a parse tree.

org.olap4j.mdx
Class AxisNode

```
java.lang.Object
  |
  +-org.olap4j.mdx.AxisNode
```

All Implemented Interfaces:
ParseTreeNode

public class **AxisNode**
extends java.lang.Object
implements ParseTreeNode

An axis in an MDX query. For example, the typical MDX query has two axes, which appear as the "ON COLUMNS" and "ON ROWS" clauses.

Constructor Summary

public	`AxisNode(ParseRegion region, boolean nonEmpty, Axis axisDef, java.util.List dimensionProperties, ParseTreeNode expression)` Creates an axis.

Method Summary

`java.lang.Object`	`accept(ParseTreeVisitor visitor)`
`AxisNode`	`deepCopy()`
`Axis`	`getAxis()` Returns the name of the axis this axis expression is populating.
`java.util.List`	`getDimensionProperties()` Returns the list of dimension properties of this axis.
`ParseTreeNode`	`getExpression()` Returns the expression which is used to compute the value of this axis.
`ParseRegion`	`getRegion()`
`Type`	`getType()`
`boolean`	`isNonEmpty()` Returns whether the axis has the `NON EMPTY` property set.
`void`	`setExpression(ParseTreeNode expr)` Sets the expression which is used to compute the value of this axis.
`void`	`setNonEmpty(boolean nonEmpty)` Sets whether the axis has the `NON EMPTY` property set.
`void`	`unparse(ParseTreeWriter writer)`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.mdx.ParseTreeNode`
`accept, deepCopy, getRegion, getType, unparse`

Constructors

AxisNode

```
public AxisNode(ParseRegion region,
                boolean nonEmpty,
                Axis axisDef,
                java.util.List dimensionProperties,
                ParseTreeNode expression)
```

Creates an axis.

Parameters:

`region` - Region of source code
`nonEmpty` - Whether to filter out members of this axis whose cells are all empty
`axisDef` - Which axis (ROWS, COLUMNS, etc.)
`dimensionProperties` - List of dimension properties; if null, empty list is assumed
`expression` - Expression to populate the axis

Methods

getRegion

```
public ParseRegion getRegion()
```

accept

```
public java.lang.Object accept(ParseTreeVisitor visitor)
```

getAxis

```
public Axis getAxis()
```

Returns the name of the axis this axis expression is populating.

Returns:

axis name

isNonEmpty

```
public boolean isNonEmpty()
```

Returns whether the axis has the `NON EMPTY` property set.

(continued from last page)

Returns:
whether the axis is NON EMPTY

setNonEmpty

`public void setNonEmpty(boolean nonEmpty)`

Sets whether the axis has the `NON EMPTY` property set. See `isNonEmpty()`.

Parameters:
`nonEmpty` - whether the axis is NON EMPTY

getExpression

`public ParseTreeNode getExpression()`

Returns the expression which is used to compute the value of this axis.

Returns:
the expression which is used to compute the value of this axis

setExpression

`public void setExpression(ParseTreeNode expr)`

Sets the expression which is used to compute the value of this axis. See `getExpression()`.

Parameters:
`expr` - the expression which is used to compute the value of this axis

unparse

`public void unparse(ParseTreeWriter writer)`

getDimensionProperties

`public java.util.List getDimensionProperties()`

Returns the list of dimension properties of this axis.

Returns:
list of dimension properties

getType

`public Type getType()`

deepCopy

`public AxisNode deepCopy()`

org.olap4j.mdx
Class CallNode

```
java.lang.Object
  |
  +-org.olap4j.mdx.CallNode
```

All Implemented Interfaces:

ParseTreeNode

public class **CallNode**
extends java.lang.Object
implements ParseTreeNode

A parse tree node representing a call to a function or operator.

Examples of calls include:

- `5 + 2`, a call to the infix arithmetic operator '+'
- `[Measures].[Unit Sales] IS NULL`, a call applying the `postfix` operator `IS NULL` to a member expression
- `CrossJoin({[Gender].Children}, {[Store]})`, a call to the `CrossJoin` function
- `[Gender].Children`, a call to the `Children` operator, which has `property syntax`
- `[Gender].Properties("FORMAT_STRING")`, a call to the `Properties` operator, which has `method syntax`

Constructor Summary

public	`CallNode(ParseRegion region, java.lang.String name, Syntax syntax, java.util.List args)` Creates a CallNode.
public	`CallNode(ParseRegion region, java.lang.String name, Syntax syntax, ParseTreeNode[] args)` Creates an CallNode using a variable number of arguments.

Method Summary

`java.lang.Object`	`accept(ParseTreeVisitor visitor)`
`CallNode`	`deepCopy()`
`java.util.List`	`getArgList()` Returns the list of arguments to this call.
`java.lang.String`	`getOperatorName()` Returns the name of the function or operator.
`ParseRegion`	`getRegion()`
`Syntax`	`getSyntax()` Returns the syntax of this call.
`Type`	`getType()`

`void`	`setType(Type type)` Sets the type of this CallNode.
`void`	`unparse(ParseTreeWriter writer)`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.mdx.ParseTreeNode`
`accept, deepCopy, getRegion, getType, unparse`

Constructors

CallNode

```
public CallNode(ParseRegion region,
                java.lang.String name,
                Syntax syntax,
                java.util.List args)
```

Creates a CallNode.

The `syntax` argument determines whether this is a prefix, infix or postfix operator, a function call, and so forth.

The list of arguments `args` must be specified, even if there are zero arguments, and each argument must be not null.

The type is initially null, but can be set using `setType(Type)` after validation.

Parameters:

`region` - Region of source code
`name` - Name of operator or function
`syntax` - Syntax of call
`args` - List of zero or more arguments

CallNode

```
public CallNode(ParseRegion region,
                java.lang.String name,
                Syntax syntax,
                ParseTreeNode[] args)
```

Creates an CallNode using a variable number of arguments.

The `syntax` argument determines whether this is a prefix, infix or postfix operator, a function call, and so forth.

The list of arguments `args` must be specified, even if there are zero arguments, and each argument must be not null.

Parameters:

`region` - Region of source code
`name` - Name of operator or function
`syntax` - Syntax of call
`args` - List of zero or more arguments

Methods

(continued from last page)

getRegion

```
public ParseRegion getRegion()
```

setType

```
public void setType(Type type)
```

Sets the type of this CallNode.

Typically, this method would be called by the validator when it has deduced the argument types, chosen between any overloaded functions or operators, and determined the result type of the function or operator.

Parameters:

`type` - Result type of this call

getType

```
public Type getType()
```

unparse

```
public void unparse(ParseTreeWriter writer)
```

accept

```
public java.lang.Object accept(ParseTreeVisitor visitor)
```

getOperatorName

```
public java.lang.String getOperatorName()
```

Returns the name of the function or operator.

Returns:

name of the function or operator

getSyntax

```
public Syntax getSyntax()
```

Returns the syntax of this call.

Returns:

the syntax of the call

getArgList

```
public java.util.List getArgList()
```

(continued from last page)

Returns the list of arguments to this call.

Returns:
list of arguments

deepCopy

```
public CallNode deepCopy()
```

org.olap4j.mdx
Class CubeNode

```
java.lang.Object
  |
  +-org.olap4j.mdx.CubeNode
```

All Implemented Interfaces:
ParseTreeNode

public class **CubeNode**
extends java.lang.Object
implements ParseTreeNode

Usage of a `Cube` as an expression in an MDX parse tree.

Constructor Summary

`public`	`CubeNode(ParseRegion region, Cube cube)` Creates a CubeNode.

Method Summary

`java.lang.Object`	`accept(ParseTreeVisitor visitor)`
`CubeNode`	`deepCopy()`
`Cube`	`getCube()` Returns the Cube used in this expression.
`ParseRegion`	`getRegion()`
`Type`	`getType()`
`java.lang.String`	`toString()`
`void`	`unparse(ParseTreeWriter writer)`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.mdx.ParseTreeNode`

`accept, deepCopy, getRegion, getType, unparse`

Constructors

(continued from last page)

CubeNode

```
public CubeNode(ParseRegion region,
                Cube cube)
```

Creates a CubeNode.

Parameters:

`region` - Region of source code
`cube` - Cube

Methods

getRegion

```
public ParseRegion getRegion()
```

getCube

```
public Cube getCube()
```

Returns the Cube used in this expression.

Returns:

cube used in this expression

accept

```
public java.lang.Object accept(ParseTreeVisitor visitor)
```

getType

```
public Type getType()
```

unparse

```
public void unparse(ParseTreeWriter writer)
```

toString

```
public java.lang.String toString()
```

deepCopy

```
public CubeNode deepCopy()
```

org.olap4j.mdx
Class DimensionNode

```
java.lang.Object
  +-org.olap4j.mdx.DimensionNode
```

All Implemented Interfaces:
ParseTreeNode

public class **DimensionNode**
extends java.lang.Object
implements ParseTreeNode

Usage of a `Dimension` as an expression in an MDX parse tree.

Constructor Summary

public	`DimensionNode(ParseRegion region, Dimension dimension)` Creates a DimensionNode.

Method Summary

java.lang.Object	`accept(ParseTreeVisitor visitor)`
DimensionNode	`deepCopy()`
Dimension	`getDimension()` Returns the Dimension used in this expression.
ParseRegion	`getRegion()`
Type	`getType()`
java.lang.String	`toString()`
void	`unparse(ParseTreeWriter writer)`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.mdx.ParseTreeNode`

`accept, deepCopy, getRegion, getType, unparse`

Constructors

(continued from last page)

DimensionNode

```
public DimensionNode(ParseRegion region,
                     Dimension dimension)
```

Creates a DimensionNode.

Parameters:

`region` - Region of source code

`dimension` - Dimension which is used in the expression

Methods

getRegion

```
public ParseRegion getRegion()
```

getDimension

```
public Dimension getDimension()
```

Returns the Dimension used in this expression.

Returns:

dimension used in this expression

accept

```
public java.lang.Object accept(ParseTreeVisitor visitor)
```

getType

```
public Type getType()
```

unparse

```
public void unparse(ParseTreeWriter writer)
```

toString

```
public java.lang.String toString()
```

deepCopy

```
public DimensionNode deepCopy()
```

org.olap4j.mdx
Class HierarchyNode

```
java.lang.Object
  |
  +-org.olap4j.mdx.HierarchyNode
```

All Implemented Interfaces:

ParseTreeNode

public class **HierarchyNode**
extends java.lang.Object
implements ParseTreeNode

Usage of a `Hierarchy` as an expression in an MDX parse tree.

Constructor Summary

`public`	`HierarchyNode(ParseRegion region, Hierarchy hierarchy)` Creates a HierarchyNode.

Method Summary

`java.lang.Object`	`accept(ParseTreeVisitor visitor)`
`HierarchyNode`	`deepCopy()`
`Hierarchy`	`getHierarchy()` Returns the Hierarchy used in this expression.
`ParseRegion`	`getRegion()`
`Type`	`getType()`
`java.lang.String`	`toString()`
`void`	`unparse(ParseTreeWriter writer)`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.mdx.ParseTreeNode`

`accept, deepCopy, getRegion, getType, unparse`

Constructors

(continued from last page)

HierarchyNode

```
public HierarchyNode(ParseRegion region,
                     Hierarchy hierarchy)
```

Creates a HierarchyNode.

Parameters:

`region` - Region of source code

`hierarchy` - Hierarchy which is used in the expression

Methods

getRegion

```
public ParseRegion getRegion()
```

getHierarchy

```
public Hierarchy getHierarchy()
```

Returns the Hierarchy used in this expression.

Returns:

hierarchy used in this expression

accept

```
public java.lang.Object accept(ParseTreeVisitor visitor)
```

getType

```
public Type getType()
```

unparse

```
public void unparse(ParseTreeWriter writer)
```

toString

```
public java.lang.String toString()
```

deepCopy

```
public HierarchyNode deepCopy()
```

org.olap4j.mdx
Class IdentifierNode

```
java.lang.Object
  |
  +-org.olap4j.mdx.IdentifierNode
```

All Implemented Interfaces:
ParseTreeNode

public class **IdentifierNode**
extends java.lang.Object
implements ParseTreeNode

Multi-part identifier.

An identifier is immutable.

Nested Class Summary

class	`IdentifierNode.Quoting` IdentifierNode.Quoting
class	`IdentifierNode.Segment` IdentifierNode.Segment

Constructor Summary

public	`IdentifierNode(IdentifierNode.Segment[] segments)` Creates an identifier containing one or more segments.
public	`IdentifierNode(java.util.List segments)` Creates an identifier containing a list of segments.

Method Summary

`java.lang.Object`	`accept(ParseTreeVisitor visitor)`
`IdentifierNode`	`append(IdentifierNode.Segment segment)` Returns a new Identifier consisting of this one with another segment appended.
`IdentifierNode`	`deepCopy()`
`ParseRegion`	`getRegion()`
`java.util.List`	`getSegmentList()` Returns the list of segments which consistitute this identifier.
`Type`	`getType()`
`static java.util.List`	`parseIdentifier(java.lang.String identifier)` Parses an MDX identifier into a list of segments.

java.lang.String	toString()
void	unparse(ParseTreeWriter writer)

Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface org.olap4j.mdx.ParseTreeNode
accept, deepCopy, getRegion, getType, unparse

Constructors

IdentifierNode

`public **IdentifierNode**(IdentifierNode.Segment[] segments)`

Creates an identifier containing one or more segments.

Parameters:
`segments` - Array of Segments, each consisting of a name and quoting style

IdentifierNode

`public **IdentifierNode**(java.util.List segments)`

Creates an identifier containing a list of segments.

Parameters:
`segments` - List of segments

Methods

getType

`public Type **getType**()`

getSegmentList

`public java.util.List **getSegmentList**()`

Returns the list of segments which consistitute this identifier.

Returns:
list of constituent segments

getRegion

`public ParseRegion **getRegion**()`

append

`public IdentifierNode **append**(IdentifierNode.Segment segment)`

Returns a new Identifier consisting of this one with another segment appended. Does not modify this Identifier.

Parameters:
`segment` - Name of segment

Returns:
New identifier

accept

`public java.lang.Object **accept**(ParseTreeVisitor visitor)`

unparse

`public void **unparse**(ParseTreeWriter writer)`

toString

`public java.lang.String **toString**()`

deepCopy

`public IdentifierNode **deepCopy**()`

parseIdentifier

`public static java.util.List **parseIdentifier**(java.lang.String identifier)`

Parses an MDX identifier into a list of segments.

Each segment is a name combined with a description of how the name was `quoted`. For example,`parseIdentifier("[Customers].USA.[South Dakota].[Sioux Falls].&[1245]")`returns `{ Segment("Customers", QUOTED), Segment("USA", UNQUOTED), Segment("South Dakota", QUOTED), Segment("Sioux Falls", QUOTED), Segment("1245", KEY) }`

Parameters:
`identifier` - MDX identifier string

Returns:
List of name segments

Throws:
`IllegalArgumentException` - if the format of the identifier is invalid

See Also:
`Cube.lookupMember(String[])`

org.olap4j.mdx
Class IdentifierNode.Segment

```
java.lang.Object
  |
  +-org.olap4j.mdx.IdentifierNode.Segment
```

public static class **IdentifierNode.Segment**
extends java.lang.Object

Component in a compound identifier. It is described by its name and how the name is quoted.

For example, the identifier `[Store].USA.[New Mexico].&[45]` has four segments:

- "Store", `QUOTED`
- "USA", `UNQUOTED`
- "New Mexico", `QUOTED`
- "45", `KEY`

To parse an identifier into a list of segments, use the method `parseIdentifier(String)`.

Constructor Summary

public	`IdentifierNode.Segment(ParseRegion region, java.lang.String name, IdentifierNode.Quoting quoting)` Creates a segment with the given quoting and region.
public	`IdentifierNode.Segment(java.lang.String name)` Creates a quoted segment, "[name]".

Method Summary

`java.lang.String`	`getName()` Returns the name of this Segment.
`IdentifierNode.Quoting`	`getQuoting()` Returns how this Segment is quoted.
`ParseRegion`	`getRegion()` Returns the region of the source code which this Segment was created from, if it was created by parsing.
`java.lang.String`	`toString()` Returns a string representation of this Segment.

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructors

(continued from last page)

IdentifierNode.Segment

```
public IdentifierNode.Segment(ParseRegion region,
                              java.lang.String name,
                              IdentifierNode.Quoting quoting)
```

Creates a segment with the given quoting and region.

Parameters:
`region` - Region of source code
`name` - Name
`quoting` - Quoting style

IdentifierNode.Segment

```
public IdentifierNode.Segment(java.lang.String name)
```

Creates a quoted segment, "[name]".

Parameters:
`name` - Name of segment

Methods

toString

```
public java.lang.String toString()
```

Returns a string representation of this Segment.

For example, "[Foo]", "&[123]", "Abc".

Returns:
String representation of this Segment

getRegion

```
public ParseRegion getRegion()
```

Returns the region of the source code which this Segment was created from, if it was created by parsing.

Returns:
region of source code

getName

```
public java.lang.String getName()
```

Returns the name of this Segment.

Returns:
name of this Segment

getQuoting

```
public IdentifierNode.Quoting getQuoting()
```

Returns how this Segment is quoted.

(continued from last page)

Returns:

how this Segment is quoted

org.olap4j.mdx
Class IdentifierNode.Quoting

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.mdx.IdentifierNode.Quoting
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

---

public static final class **IdentifierNode.Quoting**
extends java.lang.Enum

Enumeration of styles by which the component of an identifier can be quoted.

Field Summary

`public static final`	`KEY` Identifier quoted with an ampersand to indicate a key value, for example the second segment in "[Employees].&[89]".
`public static final`	`QUOTED` Quoted identifier, for example "[Measures]".
`public static final`	`UNQUOTED` Unquoted identifier, for example "Measures".

Method Summary

`static IdentifierNode.Quoting`	`valueOf(java.lang.String name)`
`static IdentifierNode.Quoting[]`	`values()`

Methods inherited from class `java.lang.Enum`
`compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `java.lang.Comparable`
`compareTo`

Fields

(continued from last page)

UNQUOTED

`public static final org.olap4j.mdx.IdentifierNode.Quoting` **`UNQUOTED`**

Unquoted identifier, for example "Measures".

QUOTED

`public static final org.olap4j.mdx.IdentifierNode.Quoting` **`QUOTED`**

Quoted identifier, for example "[Measures]".

KEY

`public static final org.olap4j.mdx.IdentifierNode.Quoting` **`KEY`**

Identifier quoted with an ampersand to indicate a key value, for example the second segment in "[Employees].&[89]".

Methods

values

`public final static IdentifierNode.Quoting[]` **`values`**`()`

valueOf

`public static IdentifierNode.Quoting` **`valueOf`**`(java.lang.String name)`

org.olap4j.mdx
Class LevelNode

```
java.lang.Object
  |
  +-org.olap4j.mdx.LevelNode
```

All Implemented Interfaces:
ParseTreeNode

public class **LevelNode**
extends java.lang.Object
implements ParseTreeNode

Usage of a `Level` as an expression in an MDX parse tree.

Constructor Summary

`public`	`LevelNode(ParseRegion region, Level level)` Creates a LevelNode.

Method Summary

`java.lang.Object`	`accept(ParseTreeVisitor visitor)`
`LevelNode`	`deepCopy()`
`Level`	`getLevel()` Returns the Level used in this expression.
`ParseRegion`	`getRegion()`
`Type`	`getType()`
`java.lang.String`	`toString()`
`void`	`unparse(ParseTreeWriter writer)`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.mdx.ParseTreeNode`

`accept, deepCopy, getRegion, getType, unparse`

Constructors

(continued from last page)

LevelNode

```
public LevelNode(ParseRegion region,
                 Level level)
```

Creates a LevelNode.

Parameters:

`region` - Region of source code

`level` - Level which is used in the expression

Methods

getRegion

```
public ParseRegion getRegion()
```

getLevel

```
public Level getLevel()
```

Returns the Level used in this expression.

Returns:

level used in this expression

accept

```
public java.lang.Object accept(ParseTreeVisitor visitor)
```

getType

```
public Type getType()
```

unparse

```
public void unparse(ParseTreeWriter writer)
```

toString

```
public java.lang.String toString()
```

deepCopy

```
public LevelNode deepCopy()
```

org.olap4j.mdx
Class LiteralNode

```
java.lang.Object
  └-org.olap4j.mdx.LiteralNode
```

All Implemented Interfaces:

ParseTreeNode

---

public class **LiteralNode**
extends java.lang.Object
implements ParseTreeNode

Represents a constant value, such as a string or number, in a parse tree.

Symbols, such as the `ASC` keyword in `Order([Store].Members, [Measures].[Unit Sales], ASC)`, are also represented as Literals.

A LiteralNode is immutable.

Method Summary

java.lang.Object	accept(ParseTreeVisitor visitor)
static LiteralNode	create(ParseRegion region, java.lang.Double value) Creates a floating-point numeric literal.
static LiteralNode	create(ParseRegion region, java.lang.Integer value) Creates an integer literal.
static LiteralNode	createNull(ParseRegion region) Creates a literal with the NULL value.
static LiteralNode	createString(ParseRegion region, java.lang.String value) Creates a string literal.
static LiteralNode	createSymbol(ParseRegion region, java.lang.String value) Creates a symbol literal.
LiteralNode	deepCopy()
ParseRegion	getRegion()
Type	getType()
java.lang.Object	getValue() Returns the value of this literal.
void	unparse(ParseTreeWriter writer)

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.mdx.ParseTreeNode`
`accept`, `deepCopy`, `getRegion`, `getType`, `unparse`

Methods

createNull

```
public static LiteralNode createNull(ParseRegion region)
```

Creates a literal with the NULL value.

Parameters:
`region` - Region of source code

Returns:
literal representing the NULL value

createString

```
public static LiteralNode createString(ParseRegion region,
          java.lang.String value)
```

Creates a string literal.

Parameters:
`region` - Region of source code
`value` - String value

Returns:
literal representing the string value

See Also:
`createSymbol(ParseRegion, String)`

createSymbol

```
public static LiteralNode createSymbol(ParseRegion region,
          java.lang.String value)
```

Creates a symbol literal.

Parameters:
`region` - Region of source code
`value` - Name of symbol

Returns:
literal representing the symbol value

See Also:
`createString(ParseRegion, String)`

(continued from last page)

create

```
public static LiteralNode create(ParseRegion region,
         java.lang.Double value)
```

Creates a floating-point numeric literal.

Parameters:

`region` - Region of source code
`value` - Value of literal; must not be null

Returns:

literal representing the floating-point value

create

```
public static LiteralNode create(ParseRegion region,
         java.lang.Integer value)
```

Creates an integer literal.

Parameters:

`region` - Region of source code
`value` - Value of literal; must not be null

Returns:

literal representing the integer value

accept

```
public java.lang.Object accept(ParseTreeVisitor visitor)
```

getType

```
public Type getType()
```

getRegion

```
public ParseRegion getRegion()
```

getValue

```
public java.lang.Object getValue()
```

Returns the value of this literal.

Returns:

value

unparse

```
public void unparse(ParseTreeWriter writer)
```

(continued from last page)

deepCopy

```
public LiteralNode deepCopy()
```

org.olap4j.mdx
Class MemberNode

```
java.lang.Object
  |
  +-org.olap4j.mdx.MemberNode
```

All Implemented Interfaces:
ParseTreeNode

public class **MemberNode**
extends java.lang.Object
implements ParseTreeNode

Usage of a `Member` as an expression in an MDX parse tree.

Constructor Summary

`public`	`MemberNode(ParseRegion region, Member member)` Creates a MemberNode.

Method Summary

`java.lang.Object`	`accept(ParseTreeVisitor visitor)`
`MemberNode`	`deepCopy()`
`Member`	`getMember()` Returns the Member used in this expression.
`ParseRegion`	`getRegion()`
`Type`	`getType()`
`java.lang.String`	`toString()`
`void`	`unparse(ParseTreeWriter writer)`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.mdx.ParseTreeNode`

`accept, deepCopy, getRegion, getType, unparse`

Constructors

(continued from last page)

MemberNode

```
public MemberNode(ParseRegion region,
                  Member member)
```

Creates a MemberNode.

Parameters:

`region` - Region of source code
`member` - Member which is used in the expression

Methods

getRegion

```
public ParseRegion getRegion()
```

getMember

```
public Member getMember()
```

Returns the Member used in this expression.

Returns:

member used in this expression

accept

```
public java.lang.Object accept(ParseTreeVisitor visitor)
```

getType

```
public Type getType()
```

unparse

```
public void unparse(ParseTreeWriter writer)
```

toString

```
public java.lang.String toString()
```

deepCopy

```
public MemberNode deepCopy()
```

org.olap4j.mdx
Class ParameterNode

```
java.lang.Object
  |
  +-org.olap4j.mdx.ParameterNode
```

All Implemented Interfaces:

ParseTreeNode

public class **ParameterNode**
extends java.lang.Object
implements ParseTreeNode

A parameter to an MDX query.

Not all dialects of MDX support parameters. If a dialect supports parameters, the driver for that dialect should extend the parser to introduce a ParameterNode into the parse tree wherever a parameter is encountered.

For example, in Mondrian's dialect of MDX, a call to the `Param(name, type, defaultValueExpr)` function introduces a parameter, and `ParamRef(name)` creates a reference to a parameter defined elsewhere in the query.

Constructor Summary

public	`ParameterNode(ParseRegion region, java.lang.String name, Type type, ParseTreeNode defaultValueExpression)` Creates a ParameterNode.

Method Summary

`java.lang.Object`	`accept(ParseTreeVisitor visitor)`
`ParameterNode`	`deepCopy()`
`ParseTreeNode`	`getDefaultValueExpression()` Returns the expression which yields the default value of this parameter.
`java.lang.String`	`getName()` Returns the name of this parameter.
`ParseRegion`	`getRegion()`
`Type`	`getType()`
`void`	`setDefaultValueExpression(ParseTreeNode defaultValueExpression)` Sets the expression which yields the default value of this parameter.
`void`	`setName(java.lang.String name)` Sets the name of this parameter.
`void`	`setType(Type type)` Sets the type of this parameter.

`void`	`unparse(ParseTreeWriter writer)`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.mdx.ParseTreeNode`
`accept, deepCopy, getRegion, getType, unparse`

Constructors

ParameterNode

```
public ParameterNode(ParseRegion region,
                     java.lang.String name,
                     Type type,
                     ParseTreeNode defaultValueExpression)
```

Creates a ParameterNode.

The `name` must not be null, and the `defaultValueExpression` must be consistent with the `type`.

Parameters:

`region` - Region of source code
`name` - Name of parameter
`type` - Type of parameter
`defaultValueExpression` - Expression which yields the default value of the parameter

Methods

getRegion

```
public ParseRegion getRegion()
```

accept

```
public java.lang.Object accept(ParseTreeVisitor visitor)
```

unparse

```
public void unparse(ParseTreeWriter writer)
```

getType

```
public Type getType()
```

(continued from last page)

getName

```
public java.lang.String getName()
```

Returns the name of this parameter.

Returns:
name of this parameter

setName

```
public void setName(java.lang.String name)
```

Sets the name of this parameter.

Parameters:
`name` - Parameter name

setType

```
public void setType(Type type)
```

Sets the type of this parameter.

Parameters:
`type` - Type

getDefaultValueExpression

```
public ParseTreeNode getDefaultValueExpression()
```

Returns the expression which yields the default value of this parameter.

Returns:
expression which yields the default value of this parameter

setDefaultValueExpression

```
public void setDefaultValueExpression(ParseTreeNode defaultValueExpression)
```

Sets the expression which yields the default value of this parameter.

Parameters:
`defaultValueExpression` - default value expression

deepCopy

```
public ParameterNode deepCopy()
```

org.olap4j.mdx
Class ParseRegion

```
java.lang.Object
  |
  +-org.olap4j.mdx.ParseRegion
```

public class **ParseRegion**
extends java.lang.Object

Region of parser source code.

The main purpose of a ParseRegion is to give detailed locations in error messages and warnings from the parsing and validation process.

A region has a start and end line number and column number. A region is a point if the start and end positions are the same.

The line and column number are one-based, because that is what end-users understand.

A region's end-points are inclusive. For example, in the code

```
SELECT FROM [Sales]
```

the `SELECT` token has region [1:1, 1:6].

Regions are immutable.

Nested Class Summary

class	`ParseRegion.RegionAndSource` ParseRegion.RegionAndSource

Constructor Summary

public	`ParseRegion(int startLine, int startColumn, int endLine, int endColumn)` Creates a ParseRegion.
public	`ParseRegion(int line, int column)` Creates a ParseRegion.

Method Summary

java.lang.String	`annotate(java.lang.String source)` Generates a string of the source code annotated with caret symbols ("^") at the beginning and end of the region.
boolean	`equals(java.lang.Object obj)`
static `ParseRegion.RegionAndSource`	`findPos(java.lang.String code)` Looks for one or two carets in an MDX string, and if present, converts them into a parser position.
int	`getEndColumn()` Return ending column number (1-based).

`int`	`getEndLine()` Return ending line number (1-based).
`int`	`getStartColumn()` Return starting column number (1-based).
`int`	`getStartLine()` Return starting line number (1-based).
`int`	`hashCode()`
`boolean`	`isPoint()` Returns whether this region has the same start and end point.
`ParseRegion`	`plusAll(java.lang.Iterable regions)` Combines this region with a list of parse tree nodes to create a region which spans from the first point in the first to the last point in the other.
`static ParseRegion`	`sum(java.lang.Iterable nodes)` Combines the parser positions of a list of nodes to create a position which spans from the beginning of the first to the end of the last.
`java.lang.String`	`toString()` Returns a string representation of this ParseRegion.

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructors

ParseRegion

```
public ParseRegion(int startLine,
                   int startColumn,
                   int endLine,
                   int endColumn)
```

Creates a ParseRegion.

All lines and columns are 1-based and inclusive. For example, the token "select" in "select from [Sales]" has a region [1:1, 1:6].

Parameters:

`startLine` - Line of the beginning of the region
`startColumn` - Column of the beginning of the region
`endLine` - Line of the end of the region
`endColumn` - Column of the end of the region

ParseRegion

```
public ParseRegion(int line,
                   int column)
```

Creates a ParseRegion. All lines and columns are 1-based.

Parameters:

(continued from last page)

`line` - Line of the beginning and end of the region
`column` - Column of the beginning and end of the region

Methods

getStartLine

`public int getStartLine()`

Return starting line number (1-based).

Returns:
1-based starting line number

getStartColumn

`public int getStartColumn()`

Return starting column number (1-based).

Returns:
1-based starting column number

getEndLine

`public int getEndLine()`

Return ending line number (1-based).

Returns:
1-based ending line number

getEndColumn

`public int getEndColumn()`

Return ending column number (1-based).

Returns:
1-based starting endings column number

toString

`public java.lang.String toString()`

Returns a string representation of this ParseRegion.

Regions are of the form `[startLine:startColumn, endLine:endColumn]`, or `[startLine:startColumn]` for point regions.

Returns:
string representation of this ParseRegion

isPoint

`public boolean isPoint()`

Returns whether this region has the same start and end point.

(continued from last page)

Returns:
whether this region has the same start and end point

hashCode

`public int hashCode()`

equals

`public boolean equals(java.lang.Object obj)`

plusAll

`public ParseRegion plusAll(java.lang.Iterable regions)`

Combines this region with a list of parse tree nodes to create a region which spans from the first point in the first to the last point in the other.

Parameters:
`regions` - Collection of source code regions

Returns:
region which represents the span of the given regions

sum

`public static ParseRegion sum(java.lang.Iterable nodes)`

Combines the parser positions of a list of nodes to create a position which spans from the beginning of the first to the end of the last.

Parameters:
`nodes` - Collection of parse tree nodes

Returns:
region which represents the span of the given nodes

findPos

`public static ParseRegion.RegionAndSource findPos(java.lang.String code)`

Looks for one or two carets in an MDX string, and if present, converts them into a parser position.

Examples:

- findPos("xxx^yyy") yields {"xxxyyy", position 3, line 1 column 4}
- findPos("xxxyyy") yields {"xxxyyy", null}
- findPos("xxx^yy^y") yields {"xxxyyy", position 3, line 4 column 4 through line 1 column 6}

Parameters:
`code` - Source code

Returns:

(continued from last page)

object containing source code annotated with region

annotate

`public java.lang.String **annotate**(java.lang.String source)`

Generates a string of the source code annotated with caret symbols ("`^`") at the beginning and end of the region.

For example, for the region `(1, 9, 1, 12)` and source `"values (foo)"`, yields the string `"values (^foo^)"`.

Parameters:

`source` - Source code

Returns:

Source code annotated with position

org.olap4j.mdx
Class ParseRegion.RegionAndSource

```
java.lang.Object
  |
  +-org.olap4j.mdx.ParseRegion.RegionAndSource
```

public static class **ParseRegion.RegionAndSource**
extends java.lang.Object

Combination of a region within an MDX statement with the source text of the whole MDX statement.

Useful for reporting errors. For example, the error in the statement

```
SELECT {[Measures].[Units In Stock]} ON COLUMNS
FROM [Sales]
```

has source "SELECT {[Measures].[Units In Stock]} ON COLUMNS\nFROM [Sales]" and region [1:9, 1:34].

Field Summary

public final	region
public final	source

Constructor Summary

public	ParseRegion.RegionAndSource(java.lang.String source, ParseRegion region) Creates a RegionAndSource.

Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Fields

source

`public final java.lang.String` **`source`**

region

`public final org.olap4j.mdx.ParseRegion` **`region`**

Constructors

(continued from last page)

ParseRegion.RegionAndSource

```
public ParseRegion.RegionAndSource(java.lang.String source,
                                   ParseRegion region)
```

Creates a RegionAndSource.

Parameters:

`source` - Source MDX code
`region` - Coordinates of region within MDX code

org.olap4j.mdx
Interface ParseTreeNode

All Known Implementing Classes:

AxisNode, CallNode, CubeNode, DimensionNode, HierarchyNode, IdentifierNode, LevelNode, LiteralNode, MemberNode, ParameterNode, PropertyValueNode, SelectNode, WithMemberNode, WithSetNode

public interface **ParseTreeNode**
extends

Node in a parse tree representing a parsed MDX statement.

To convert a parse tree to an MDX string, use a `ParseTreeWriter` and the `unparse(ParseTreeWriter)` method.

Method Summary

`java.lang.Object`	`accept(ParseTreeVisitor visitor)` Accepts a visitor to this MDX parse tree node.
`ParseTreeNode`	`deepCopy()` Creates a deep copy of this ParseTreeNode object.
`ParseRegion`	`getRegion()` Returns the region of the source code which this node was created from, if it was created by parsing.
`Type`	`getType()` Returns the type of this expression.
`void`	`unparse(ParseTreeWriter writer)` Converts this node into MDX text.

Methods

accept

```
public java.lang.Object accept(ParseTreeVisitor visitor)
```

Accepts a visitor to this MDX parse tree node.

The implementation should generally dispatches to the `visit` method appropriate to the type of expression.

Parameters:

`visitor` - Visitor

Returns:

T, the specific return type of the visitor

getType

```
public Type getType()
```

(continued from last page)

Returns the type of this expression.

Returns null if this node is not an expression, for instance a `SELECT` node.

Returns:
type of this expression

unparse

`public void unparse(ParseTreeWriter writer)`

Converts this node into MDX text.

Parameters:
`writer` - Parse tree writer

getRegion

`public ParseRegion getRegion()`

Returns the region of the source code which this node was created from, if it was created by parsing.

A non-leaf node's region will encompass the regions of all of its children. For example, a the region of a function call node `Crossjoin([Gender], {[Store].[USA]})` stretches from the first character of the function name to the closing parenthesis.

Region may be null, if the node was created programmatically, not from a piece of source code.

Returns:
Region of the source code this node was created from, if it was created by parsing

deepCopy

`public ParseTreeNode deepCopy()`

Creates a deep copy of this ParseTreeNode object.

Note: implementing classes can return the concrete type instead of ParseTreeNode (using Java 1.5 covariant return types)

Returns:
The deep copy of this ParseTreeNode

org.olap4j.mdx
Interface ParseTreeVisitor

public interface **ParseTreeVisitor**
extends

Interface for a visitor to an MDX parse tree.

Together with the `ParseTreeNode.accept(ParseTreeVisitor)` method, an class implementing this interface implements a visitor pattern, to allow an algorithm to efficiently traverse a parse tree and perform an action at each node dependent upon the type of each node.

Method Summary

`java.lang.Object`	`visit(AxisNode axis)` Visits an axis of a select statement.
`java.lang.Object`	`visit(CallNode call)` Visits a call to an operator or function.
`java.lang.Object`	`visit(CubeNode cubeNode)` Visits a use of a `Cube` in a select statement.
`java.lang.Object`	`visit(DimensionNode dimensionNode)` Visits a use of a `Dimension` in a select statement.
`java.lang.Object`	`visit(HierarchyNode hierarchyNode)` Visits a use of a `Hierarchy` in a select statement.
`java.lang.Object`	`visit(IdentifierNode id)` Visits an identifier.
`java.lang.Object`	`visit(LevelNode levelNode)` Visits a use of a `Level` in a select statement.
`java.lang.Object`	`visit(LiteralNode literalNode)` Visits a literal.
`java.lang.Object`	`visit(MemberNode memberNode)` Visits a use of a `Member` in a select statement.
`java.lang.Object`	`visit(ParameterNode parameterNode)` Visits a parameter.
`java.lang.Object`	`visit(PropertyValueNode propertyValueNode)` Visits a property-value pair.
`java.lang.Object`	`visit(SelectNode selectNode)` Visits a select statement.
`java.lang.Object`	`visit(WithMemberNode calcMemberNode)` Visits a member declaration.
`java.lang.Object`	`visit(WithSetNode calcSetNode)` Visits a set declaration.

Methods

visit

`public java.lang.Object **visit**(SelectNode selectNode)`

Visits a select statement.

Parameters:
`selectNode` - Node representing a select statement

Returns:
value yielded by visiting the node

See Also:
`accept(ParseTreeVisitor)`

visit

`public java.lang.Object **visit**(AxisNode axis)`

Visits an axis of a select statement.

Parameters:
`axis` - Node representing an axis

Returns:
value yielded by visiting the node

See Also:
`accept(ParseTreeVisitor)`

visit

`public java.lang.Object **visit**(WithMemberNode calcMemberNode)`

Visits a member declaration.

Parameters:
`calcMemberNode` - Node representing a member declaration

Returns:
value yielded by visiting the node

See Also:
`accept(ParseTreeVisitor)`

visit

`public java.lang.Object **visit**(WithSetNode calcSetNode)`

Visits a set declaration.

Parameters:
`calcSetNode` - Node representing a set declaration

(continued from last page)

Returns:
value yielded by visiting the node

See Also:
`accept(ParseTreeVisitor)`

visit

`public java.lang.Object **visit**(CallNode call)`

Visits a call to an operator or function.

Parameters:
`call` - Node representing a call to an operator or function

Returns:
value yielded by visiting the node

See Also:
`accept(ParseTreeVisitor)`

visit

`public java.lang.Object **visit**(IdentifierNode id)`

Visits an identifier.

Parameters:
`id` - Node representing an identifier

Returns:
value yielded by visiting the node

See Also:
`accept(ParseTreeVisitor)`

visit

`public java.lang.Object **visit**(ParameterNode parameterNode)`

Visits a parameter.

Parameters:
`parameterNode` - Node representing use of a parameter

Returns:
value yielded by visiting the node

See Also:
`accept(ParseTreeVisitor)`

visit

`public java.lang.Object **visit**(CubeNode cubeNode)`

Visits a use of a `Cube` in a select statement.

Parameters:
`cubeNode` - Node representing a use of a Cube

(continued from last page)

Returns:
value yielded by visiting the node

See Also:
`accept(ParseTreeVisitor)`

visit

`public java.lang.Object visit(DimensionNode dimensionNode)`

Visits a use of a `Dimension` in a select statement.

Parameters:
`dimensionNode` - Node representing a use of a Dimension

Returns:
value yielded by visiting the node

See Also:
`accept(ParseTreeVisitor)`

visit

`public java.lang.Object visit(HierarchyNode hierarchyNode)`

Visits a use of a `Hierarchy` in a select statement.

Parameters:
`hierarchyNode` - Node representing a use of a Hierarchy

Returns:
value yielded by visiting the node

See Also:
`accept(ParseTreeVisitor)`

visit

`public java.lang.Object visit(LevelNode levelNode)`

Visits a use of a `Level` in a select statement.

Parameters:
`levelNode` - Node representing a use of a Level

Returns:
value yielded by visiting the node

See Also:
`accept(ParseTreeVisitor)`

visit

`public java.lang.Object visit(MemberNode memberNode)`

Visits a use of a `Member` in a select statement.

Parameters:

(continued from last page)

`memberNode` - Node representing a use of a Member

Returns:

value yielded by visiting the node

See Also:

`accept(ParseTreeVisitor)`

visit

`public java.lang.Object visit(LiteralNode literalNode)`

Visits a literal.

Parameters:

`literalNode` - Node representing a Literal

Returns:

value yielded by visiting the node

See Also:

`accept(ParseTreeVisitor)`

visit

`public java.lang.Object visit(PropertyValueNode propertyValueNode)`

Visits a property-value pair.

Parameters:

`propertyValueNode` - Node representing a property-value pair

Returns:

value yielded by visiting the node

See Also:

`accept(ParseTreeVisitor)`

org.olap4j.mdx
Class ParseTreeWriter

```
java.lang.Object
  |
  +-org.olap4j.mdx.ParseTreeWriter
```

public class **ParseTreeWriter**
extends java.lang.Object

Writer for MDX parse tree.

Typical use is with the `unparse(ParseTreeWriter)` method as follows:

```
ParseTreeNode node;
StringWriter sw = new StringWriter();
PrintWriter pw = new PrintWriter(sw);
ParseTreeWriter mdxWriter = new ParseTreeWriter(pw);
node.unparse(mdxWriter);
pw.flush();
String mdx = sw.toString();
```

See Also:
`ParseTreeNode.unparse(ParseTreeWriter)`

Constructor Summary

`public`	`ParseTreeWriter(java.io.PrintWriter pw)` Creates a ParseTreeWriter.

Method Summary

`java.io.PrintWriter`	`getPrintWriter()` Returns the underlying writer.

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructors

ParseTreeWriter

`public ParseTreeWriter(java.io.PrintWriter pw)`

Creates a ParseTreeWriter.

Parameters:
`pw` - Underlying writer

(continued from last page)

Methods

getPrintWriter

```
public java.io.PrintWriter getPrintWriter()
```

Returns the underlying writer.

Returns:
underlying writer

org.olap4j.mdx
Class PropertyValueNode

```
java.lang.Object
  |
  +-org.olap4j.mdx.PropertyValueNode
```

All Implemented Interfaces:

ParseTreeNode

public class **PropertyValueNode**
extends java.lang.Object
implements ParseTreeNode

Parse tree node representing a property-value pair.

Property-value pairs are used to define properties of calculated members. For example, in `WITH MEMBER [Measures].[Foo] AS ' [Measures].[Unit Sales] ', FORMAT_STRING = 'Bold', SOLVE_ORDER = 2 SELECT ...` there are two property-value pairs FORMAT_STRING and SOLVE_ORDER.

Constructor Summary

public	`PropertyValueNode(ParseRegion region, java.lang.String name, ParseTreeNode expression)` Creates a PropertyValueNode.

Method Summary

`java.lang.Object`	`accept(ParseTreeVisitor visitor)`
`PropertyValueNode`	`deepCopy()`
`ParseTreeNode`	`getExpression()` Returns the expression by which the value of the property is derived.
`java.lang.String`	`getName()` Returns the name of the property
`ParseRegion`	`getRegion()`
`Type`	`getType()`
`void`	`unparse(ParseTreeWriter writer)`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.mdx.ParseTreeNode`

`accept, deepCopy, getRegion, getType, unparse`

Constructors

PropertyValueNode

```
public PropertyValueNode(ParseRegion region,
                         java.lang.String name,
                         ParseTreeNode expression)
```

Creates a PropertyValueNode.

Parameters:
`region` - Region of source code
`name` - Name of property
`expression` - Expression for value of property (often a literal)

Methods

getRegion

```
public ParseRegion getRegion()
```

getType

```
public Type getType()
```

getExpression

```
public ParseTreeNode getExpression()
```

Returns the expression by which the value of the property is derived.

Returns:
the expression by which the value of the property is derived

getName

```
public java.lang.String getName()
```

Returns the name of the property

Returns:
name of the property

accept

```
public java.lang.Object accept(ParseTreeVisitor visitor)
```

unparse

```
public void unparse(ParseTreeWriter writer)
```

(continued from last page)

deepCopy

```
public PropertyValueNode deepCopy()
```

org.olap4j.mdx
Class SelectNode

```
java.lang.Object
  |
  +-org.olap4j.mdx.SelectNode
```

All Implemented Interfaces:

ParseTreeNode

public class **SelectNode**
extends java.lang.Object
implements ParseTreeNode

Parse tree model for an MDX SELECT statement.

Constructor Summary

`public`	`SelectNode(ParseRegion region, java.util.List withList, java.util.List axisList, ParseTreeNode from, AxisNode filterAxis, java.util.List cellPropertyList)` Creates a SelectNode.
`public`	`SelectNode()` Creates an empty SelectNode.

Method Summary

`java.lang.Object`	`accept(ParseTreeVisitor visitor)`
`SelectNode`	`deepCopy()`
`java.util.List`	`getAxisList()` Returns a list of axes in this SelectNode.
`java.util.List`	`getCellPropertyList()` Returns a list of cell properties in this SelectNode.
`AxisNode`	`getFilterAxis()` Returns the filter axis defined by the WHERE clause of this SelectNode, or null if there is no filter axis.
`ParseTreeNode`	`getFrom()` Returns the node representing the FROM clause of this SELECT statement.
`ParseRegion`	`getRegion()`
`Type`	`getType()`
`java.util.List`	`getWithList()` Returns a list of calculated members and sets defined as the WITH clause of this SelectNode.

`void`	`setFrom(ParseTreeNode fromNode)` Sets the FROM clause of this SELECT statement.
`java.lang.String`	`toString()`
`void`	`unparse(ParseTreeWriter writer)`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.mdx.ParseTreeNode`

`accept, deepCopy, getRegion, getType, unparse`

Constructors

SelectNode

```
public SelectNode(ParseRegion region,
                  java.util.List withList,
                  java.util.List axisList,
                  ParseTreeNode from,
                  AxisNode filterAxis,
                  java.util.List cellPropertyList)
```

Creates a SelectNode.

Parameters:

`region` - Region of source code from which this node was created
`withList` - List of members and sets defined in this query using a `WITH` clause
`axisList` - List of axes
`from` - Name of cube
`filterAxis` - Filter axis
`cellPropertyList` - List of properties

SelectNode

```
public SelectNode()
```

Creates an empty SelectNode.

The contents of the SelectNode, such as the axis list, can be populated after construction.

Methods

getRegion

```
public ParseRegion getRegion()
```

accept

```
public java.lang.Object accept(ParseTreeVisitor visitor)
```

(continued from last page)

getType

`public Type getType()`

toString

`public java.lang.String toString()`

unparse

`public void unparse(ParseTreeWriter writer)`

getWithList

`public java.util.List getWithList()`

Returns a list of calculated members and sets defined as the WITH clause of this SelectNode.

For example, the WITH clause of query`WITH MEMBER [Measures].[Foo] AS ' [Measures].[Unit Sales] * 2 ' SET [Customers].[Top] AS ' TopCount([Customers].Members, 10) ' SELECT FROM [Sales]`contains one `WithMemberNode` and one `WithSetNode`.

The returned list is mutable.

Returns:
list of calculated members and sets

getAxisList

`public java.util.List getAxisList()`

Returns a list of axes in this SelectNode.

The returned list is mutable.

Returns:
list of axes

getFilterAxis

`public AxisNode getFilterAxis()`

Returns the filter axis defined by the WHERE clause of this SelectNode, or null if there is no filter axis.

Returns:
filter axis

getFrom

`public ParseTreeNode getFrom()`

(continued from last page)

Returns the node representing the FROM clause of this SELECT statement. The node is typically an `IdentifierNode` or a `CubeNode`.

Returns:
FROM clause

setFrom

`public void setFrom(ParseTreeNode fromNode)`

Sets the FROM clause of this SELECT statement.

`fromNode` should typically by an `IdentifierNode` containing the cube name, or a `CubeNode` referencing an explicit `Cube` object.

Parameters:
`fromNode` - FROM clause

getCellPropertyList

`public java.util.List getCellPropertyList()`

Returns a list of cell properties in this SelectNode.

The returned list is mutable.

Returns:
list of cell properties

deepCopy

`public SelectNode deepCopy()`

org.olap4j.mdx
Class Syntax

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.mdx.Syntax
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

public class **Syntax**
extends java.lang.Enum

Enumerated values describing the syntax of an expression.

Field Summary

`public static final`	`AmpersandQuotedProperty` Defines syntax for expression invoked `object.[&PROPERTY]` (a variant of `Property`).
`public static final`	`Braces` Defines syntax for expression invoked as `{ARG, ...}`; that is, the set construction operator.
`public static final`	`Case` Defines syntax for expression invoked as `CASE ...`
`public static final`	`Cast` Defines syntax for a CAST expression `CAST(expression AS type)`.
`public static final`	`Function` Defines syntax for expression invoked `FUNCTION()` or `FUNCTION(args)`.
`public static final`	`Infix` Defines syntax for expression invoked as `arg OPERATOR arg` (like '+' or 'AND').
`public static final`	`Internal` Defines syntax for expression generated by the system which cannot be specified syntactically.
`public static final`	`Method` Defines syntax for expression invoked invoked as `object.METHOD()` or `object.METHOD(args)`.
`public static final`	`Parentheses` Defines syntax for expression invoked as `(ARG)` or `(ARG, ...)`; that is, parentheses for grouping expressions, and the tuple construction operator.
`public static final`	`Postfix` Defines syntax for expression invoked as `arg OPERATOR` (like `IS EMPTY`).
`public static final`	`Prefix` Defines syntax for expression invoked as `OPERATOR arg` (like unary '-').

public static final	Property Defines syntax for expression invoked as `object.PROPERTY`.
public static final	QuotedProperty Defines syntax for expression invoked `object.&PROPERTY` (a variant of `Property`).

Method Summary

void	`unparse(java.lang.String operatorName, java.util.List argList, ParseTreeWriter writer)` Converts a call to a function of this syntax into source code.
static Syntax	`valueOf(java.lang.String name)`
static Syntax[]	`values()`

Methods inherited from class `java.lang.Enum`

`compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `java.lang.Comparable`

`compareTo`

Fields

Function

`public static final org.olap4j.mdx.Syntax` **`Function`**

Defines syntax for expression invoked `FUNCTION()` or `FUNCTION(args)`.

Property

`public static final org.olap4j.mdx.Syntax` **`Property`**

Defines syntax for expression invoked as `object.PROPERTY`.

Method

`public static final org.olap4j.mdx.Syntax` **`Method`**

Defines syntax for expression invoked invoked as `object.METHOD()` or `object.METHOD(args)`.

Infix

`public static final org.olap4j.mdx.Syntax` **`Infix`**

Defines syntax for expression invoked as `arg OPERATOR arg` (like '+' or 'AND').

Prefix

`public static final org.olap4j.mdx.Syntax **Prefix**`

Defines syntax for expression invoked as `OPERATOR arg` (like unary '-').

Postfix

`public static final org.olap4j.mdx.Syntax **Postfix**`

Defines syntax for expression invoked as `arg OPERATOR` (like `IS EMPTY`).

Braces

`public static final org.olap4j.mdx.Syntax **Braces**`

Defines syntax for expression invoked as `{ARG, ...}`; that is, the set construction operator.

Parentheses

`public static final org.olap4j.mdx.Syntax **Parentheses**`

Defines syntax for expression invoked as `(ARG)` or `(ARG, ...)`; that is, parentheses for grouping expressions, and the tuple construction operator.

Case

`public static final org.olap4j.mdx.Syntax **Case**`

Defines syntax for expression invoked as `CASE ... END`.

Internal

`public static final org.olap4j.mdx.Syntax **Internal**`

Defines syntax for expression generated by the system which cannot be specified syntactically.

Cast

`public static final org.olap4j.mdx.Syntax **Cast**`

Defines syntax for a CAST expression `CAST(expression AS type)`.

QuotedProperty

`public static final org.olap4j.mdx.Syntax **QuotedProperty**`

Defines syntax for expression invoked `object.&PROPERTY` (a variant of `Property`).

AmpersandQuotedProperty

`public static final org.olap4j.mdx.Syntax **AmpersandQuotedProperty**`

Defines syntax for expression invoked `object.[&PROPERTY]` (a variant of `Property`).

Methods

(continued from last page)

values

```
public final static Syntax[] values()
```

valueOf

```
public static Syntax valueOf(java.lang.String name)
```

unparse

```
public void unparse(java.lang.String operatorName,
         java.util.List argList,
         ParseTreeWriter writer)
```

Converts a call to a function of this syntax into source code.

Parameters:

`operatorName` - Operator name
`argList` - List of arguments
`writer` - Writer

org.olap4j.mdx
Class WithMemberNode

```
java.lang.Object
  └-org.olap4j.mdx.WithMemberNode
```

All Implemented Interfaces:
ParseTreeNode

public class **WithMemberNode**
extends java.lang.Object
implements ParseTreeNode

Parse tree node which declares a calculated member. Represented as the `WITH MEMBER` clause of an MDX `SELECT` statement.

Constructor Summary

`public`	`WithMemberNode(ParseRegion region, IdentifierNode name, ParseTreeNode exp, java.util.List memberPropertyList)` Constructs a formula specifying a member.

Method Summary

`java.lang.Object`	`accept(ParseTreeVisitor visitor)`
`WithMemberNode`	`deepCopy()`
`ParseTreeNode`	`getExpression()` Returns the expression to evaluate to calculate the member.
`IdentifierNode`	`getIdentifier()` Returns the name of the member declared.
`java.util.List`	`getMemberPropertyList()` Returns the list of properties of this member.
`ParseRegion`	`getRegion()`
`Type`	`getType()`
`void`	`setExpression(ParseTreeNode expression)` Sets the expression to evaluate to calculate the member.
`void`	`unparse(ParseTreeWriter writer)`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.mdx.ParseTreeNode`

`accept, deepCopy, getRegion, getType, unparse`

Constructors

WithMemberNode

```
public WithMemberNode(ParseRegion region,
                      IdentifierNode name,
                      ParseTreeNode exp,
                      java.util.List memberPropertyList)
```

Constructs a formula specifying a member.

Parameters:

`region` - Source code region
`name` - Name of member being declared
`exp` - Expression for value of member
`memberPropertyList` - Collection of properties of member

Methods

getRegion

```
public ParseRegion getRegion()
```

unparse

```
public void unparse(ParseTreeWriter writer)
```

getIdentifier

```
public IdentifierNode getIdentifier()
```

Returns the name of the member declared.

The name is as specified in the parse tree; it may not be identical to the unique name of the member.

Returns:

Name of member

getExpression

```
public ParseTreeNode getExpression()
```

Returns the expression to evaluate to calculate the member.

Returns:

expression

setExpression

```
public void setExpression(ParseTreeNode expression)
```

Sets the expression to evaluate to calculate the member.

(continued from last page)

Parameters:
`expression` - Expression

accept

`public java.lang.Object accept(ParseTreeVisitor visitor)`

getType

`public Type getType()`

getMemberPropertyList

`public java.util.List getMemberPropertyList()`

Returns the list of properties of this member.

The list may be empty, but is never null. Each entry is a (name, expression) pair.

Returns:
list of properties

deepCopy

`public WithMemberNode deepCopy()`

org.olap4j.mdx
Class WithSetNode

```
java.lang.Object
  |
  +-org.olap4j.mdx.WithSetNode
```

All Implemented Interfaces:
ParseTreeNode

public class **WithSetNode**
extends java.lang.Object
implements ParseTreeNode

Parse tree node which declares a calculated set. Represented as the `WITH SET` clause of an MDX `SELECT` statement.

Constructor Summary

`public`	`WithSetNode(ParseRegion region, IdentifierNode name, ParseTreeNode expression)` Creates a declaration of a named set.

Method Summary

`java.lang.Object`	`accept(ParseTreeVisitor visitor)`
`WithSetNode`	`deepCopy()`
`ParseTreeNode`	`getExpression()` Returns the expression which calculates the set.
`IdentifierNode`	`getIdentifier()` Returns the name of the set.
`ParseRegion`	`getRegion()`
`Type`	`getType()`
`void`	`setExpression(ParseTreeNode expression)` Sets the expression which calculates the set.
`void`	`unparse(ParseTreeWriter writer)`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.mdx.ParseTreeNode`
`accept, deepCopy, getRegion, getType, unparse`

(continued from last page)

Constructors

WithSetNode

```
public WithSetNode(ParseRegion region,
                   IdentifierNode name,
                   ParseTreeNode expression)
```

Creates a declaration of a named set.

Parameters:

`region` - Region of source code
`name` - Name of set
`expression` - Expression to calculate set

Methods

getRegion

```
public ParseRegion getRegion()
```

unparse

```
public void unparse(ParseTreeWriter writer)
```

getIdentifier

```
public IdentifierNode getIdentifier()
```

Returns the name of the set.

Returns:

name of the set

getExpression

```
public ParseTreeNode getExpression()
```

Returns the expression which calculates the set.

Returns:

expression which calculates the set

setExpression

```
public void setExpression(ParseTreeNode expression)
```

Sets the expression which calculates the set.

Parameters:

`expression` - expression which calculates the set

(continued from last page)

accept

```
public java.lang.Object accept(ParseTreeVisitor visitor)
```

getType

```
public Type getType()
```

deepCopy

```
public WithSetNode deepCopy()
```

Package
org.olap4j.mdx.parser

Provides an API for parsing statements and expressions in the MDX language.

org.olap4j.mdx.parser
Class MdxParseException

```
java.lang.Object
  |
  +-java.lang.Throwable
      |
      +-java.lang.Exception
          |
          +-java.lang.RuntimeException
              |
              +-org.olap4j.mdx.parser.MdxParseException
```

All Implemented Interfaces:
java.io.Serializable

---

public class **MdxParseException**
extends java.lang.RuntimeException

Exception thrown by an `MdxParser` to indicate an error in parsing. Has a `ParseRegion`.

Constructor Summary

`public`	`MdxParseException(ParseRegion region, java.lang.Throwable cause)` Creates an MdxParseException with a region of the source code and a specified cause.
`public`	`MdxParseException(ParseRegion region, java.lang.String message)` Creates an MdxParseException with a region of the source code and a specified detail message.

Method Summary

`ParseRegion`	`getRegion()`

Methods inherited from class `java.lang.Throwable`

`fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructors

MdxParseException

```
public MdxParseException(ParseRegion region,
                         java.lang.Throwable cause)
```

Creates an MdxParseException with a region of the source code and a specified cause.

Parameters:
`region` - Region of source code which contains the error

(continued from last page)

`cause` - the cause (which is saved for later retrieval by the `Throwable.getCause()` method). (A null value is permitted, and indicates that the cause is nonexistent or unknown.)

MdxParseException

```
public MdxParseException(ParseRegion region,
                         java.lang.String message)
```

Creates an MdxParseException with a region of the source code and a specified detail message.

Parameters:

`region` - Region of source code which contains the error

`message` - the detail message. The detail message is saved for later retrieval by the `Throwable.getMessage()` method.

Methods

getRegion

```
public ParseRegion getRegion()
```

org.olap4j.mdx.parser
Interface MdxParser

public interface **MdxParser**
extends

Parser for the MDX query language.

A parser is reusable but not reentrant: you can call `parseSelect(String)` and `parseExpression(String)` several times, but not at the same time from different threads.
See Also:
`MdxParserFactory`

Method Summary

`ParseTreeNode`	`parseExpression(java.lang.String mdx)` Parses an MDX expression and returns a parse tree.
`SelectNode`	`parseSelect(java.lang.String mdx)` Parses an MDX Select statement and returns the `SelectNode` at the root of the parse tree.

Methods

parseSelect

```
public SelectNode parseSelect(java.lang.String mdx)
```

Parses an MDX Select statement and returns the `SelectNode` at the root of the parse tree.

In order to be parsed successfully, the expression must be syntactically correct but does not need to be valid. (Syntactic correctness and validity are described further in the description of `parseExpression(String)`.)

Parameters:
`mdx` - MDX query string

Returns:
Parse tree

parseExpression

```
public ParseTreeNode parseExpression(java.lang.String mdx)
```

Parses an MDX expression and returns a parse tree.

An expression is a combination of operators and operands, which can occur in many places inside an MDX query, such as the definition of a calculated member or an axis.

In order to be parsed successfully, the expression must be syntactically correct but does not need to be valid. For example,`(1 + (2 + 3)`is syntactically incorrect, because there are more open parentheses "(" than close parentheses ")", and the parser will give an error. Conversely, `(1 + [Measures].[Bad Measure])`is syntactically correct, and the parser will successfully create a parse tree, even if `[Measures].[Bad Measure]` does not exist.

Parameters:
`mdx` - MDX expression

(continued from last page)

Returns:

Parse tree

org.olap4j.mdx.parser
Interface MdxParserFactory

public interface **MdxParserFactory**
extends

Factory for MDX parsers.

Method Summary

`MdxParser`	`createMdxParser(OlapConnection connection)` Creates an MDX parser.
`MdxValidator`	`createMdxValidator(OlapConnection connection)` Creates an MDX validator.

Methods

createMdxParser

```
public MdxParser createMdxParser(OlapConnection connection)
```

Creates an MDX parser.

Parameters:
`connection` - Connection in which to resolve identifiers

Returns:
MDX parser

createMdxValidator

```
public MdxValidator createMdxValidator(OlapConnection connection)
```

Creates an MDX validator.

Parameters:
`connection` - Connection in which to resolve identifiers

Returns:
MDX validator

org.olap4j.mdx.parser
Interface MdxValidator

public interface **MdxValidator**
extends

Validator for the MDX query language.

A validator is reusable but not reentrant: you can call `validateSelect(SelectNode)` several times, but not at the same time from different threads.

To create a validator, use the `createMdxValidator(org.olap4j.OlapConnection)` method.
See Also:
`MdxParserFactory`, `MdxParser`

Method Summary

`SelectNode`	`validateSelect(SelectNode selectNode)` Validates an MDX SELECT statement.

Methods

validateSelect

```
public SelectNode validateSelect(SelectNode selectNode)
  throws OlapException
```

Validates an MDX SELECT statement.

The SelectNode representing the SELECT statement may have been created by an `MdxParser`, or it may have been built programmatically.

If the parse tree is invalid, throws an `OlapException`.

If it is valid, returns a parse tree. This parse tree may or may not be the same parse tree passed as an argument. After validation, you can ascertain the type of each node of the parse tree by calling its `ParseTreeNode.getType()` method.

Parameters:
`selectNode` - Parse tree node representing a SELECT statement

Returns:
Validated parse tree

Throws:
`OlapException` - if node is invalid

Package
org.olap4j.metadata

Provides classes and interfaces for browsing an OLAP schema.

Schemas have a hierarchical structure:

- `OlapConnection`
  - `Catalog`
    - `Schema`
      - `Cube`
        - `Dimension`
          - `Hierarchy`
            - `Level`
              - `Member`
              - `Property`
        - `NamedSet`
      - Dimension (shared)

org.olap4j.metadata
Interface Catalog

public interface **Catalog**
extends

Highest level element in the hierarchy of metadata objects.

A Catalog contains one or more `Schema`s.

Some OLAP servers may only have one Catalog. Mondrian is one such OLAP server; its sole catalog is called "LOCALDB".

To obtain the collection of catalogs in the current server, call the `OlapConnection.getCatalogs()` method.

The hierarchy of metadata objects, rooted at the connection from which they are accessed, is as follows:

- `OlapConnection`
  - `Catalog`
    - `Schema`
      - `Cube`
        - `Dimension`
          - `Hierarchy`
            - `Level`
              - `Member`
              - `Property`
        - `NamedSet`
      - Dimension (shared)

Method Summary

`OlapDatabaseMetaData`	`getMetaData()` Retrieves the metadata describing the OLAP server that this Catalog belongs to.
`java.lang.String`	`getName()` Returns the name of this Catalog.
`NamedList`	`getSchemas()` Returns a list of `Schema` objects which belong to this `Catalog`.

Methods

getSchemas

```
public NamedList getSchemas()
  throws OlapException
```

Returns a list of `Schema` objects which belong to this `Catalog`.

The caller should assume that the list is immutable; if the caller modifies the list, behavior is undefined.

Returns:
List of Schema in this `Catalog`

Throws:
`OlapException` - if error occurs

(continued from last page)

See Also:

`DatabaseMetaData.getSchemas()`

getName

```
public java.lang.String getName()
```

Returns the name of this Catalog.

Returns:

name of this Catalog

getMetaData

```
public OlapDatabaseMetaData getMetaData()
```

Retrieves the metadata describing the OLAP server that this Catalog belongs to.

Returns:

metadata describing the OLAP server

org.olap4j.metadata
Interface Cube

All Superinterfaces:
MetadataElement

public interface **Cube**
extends MetadataElement

Central metadata object for representation of multidimensional data.

A Cube belongs to a `Schema`, and is described by a list of `Dimension`s and a list of `Measure`s. It may also have one or more `NamedSet`s.

See Also:
`getMeasures()`

Method Summary

`NamedList`	`getDimensions()` Returns a list of `Dimension` objects in this Cube.
`NamedList`	`getHierarchies()` Returns a list of `Hierarchy` objects in this Cube.
`java.util.List`	`getMeasures()` Returns a list of `Measure` objects in this Cube.
`Schema`	`getSchema()` Returns the `Schema` this Cube belongs to.
`NamedList`	`getSets()` Returns a list of `NamedSet` objects in this Cube.
`java.util.Collection`	`getSupportedLocales()` Returns a collection of `java.util.Locale` objects for which this `Cube` has been localized.
`Member`	`lookupMember(java.lang.String[] nameParts)` Finds a member in the current Cube based upon its fully-qualified name.
`java.util.List`	`lookupMembers(java.util.Set treeOps, java.lang.String[] nameParts)` Finds a collection of members in the current Cube related to a given member.

Methods inherited from interface `org.olap4j.metadata.MetadataElement`

`getCaption`, `getDescription`, `getName`, `getUniqueName`

Methods

getSchema

```
public Schema getSchema()
```

Returns the `Schema` this Cube belongs to.

(continued from last page)

Returns:
Schema this Cube belongs to

getDimensions

public NamedList **getDimensions**()

Returns a list of `Dimension` objects in this Cube.

The caller should assume that the list is immutable; if the caller modifies the list, behavior is undefined.

Returns:
list of Dimensions

See Also:
`OlapDatabaseMetaData.getDimensions(String, String, String, String)`

getHierarchies

public NamedList **getHierarchies**()

Returns a list of `Hierarchy` objects in this Cube.

The caller should assume that the list is immutable; if the caller modifies the list, behavior is undefined.

Returns:
list of Dimensions

See Also:
`OlapDatabaseMetaData.getHierarchies(String, String, String, String, String)`

getMeasures

public java.util.List **getMeasures**()

Returns a list of `Measure` objects in this Cube.

The list includes both stored and calculated members, and (unlike the `OlapDatabaseMetaData.getMeasures(String, String, String, String, String)` method or the MDSCHEMA_MEASURES XMLA request) is sorted by ordinal.

Returns:
list of Measures

See Also:
`OlapDatabaseMetaData.getMeasures(String, String, String, String, String)`

getSets

public NamedList **getSets**()

Returns a list of `NamedSet` objects in this Cube.

The caller should assume that the list is immutable; if the caller modifies the list, behavior is undefined.

Returns:
list of NamedSets

See Also:

(continued from last page)

`OlapDatabaseMetaData.getSets(String, String, String, String)`

getSupportedLocales

```
public java.util.Collection getSupportedLocales()
```

Returns a collection of `java.util.Locale` objects for which this `Cube` has been localized.

Consider the following use case. Suppose one cube is available in English and French, and in French and Spanish, and both are shown in same portal. Clients typically say that seeing reports in a mixture of languages is confusing; the portal would figure out the best common language, in this case French. This method allows the client to choose the most appropriate locale.

The list is advisory: a client is free to choose another locale, in which case, the server will probably revert to the base locale for locale-specific behavior such as captions and formatting.

Returns:

List of locales for which this `Cube` has been localized

See Also:

`getSupportedLocales`

lookupMember

```
public Member lookupMember(java.lang.String[] nameParts)
  throws OlapException
```

Finds a member in the current Cube based upon its fully-qualified name. Returns the member, or null if there is no member with this name.

The fully-qualified name starts with the name of the dimension, followed by the name of a root member, and continues with the name of each successive member on the path from the root member. If a member's name is unique within its level, preceding member name can be omitted.

For example, `lookupMember("Product", "Food")` and `lookupMember("Product", "All Products", "Food")` are both valid ways to locate the "Food" member of the "Product" dimension.

Parameters:

`nameParts` - Components of the fully-qualified member name

Returns:

member with the given name, or null if not found

Throws:

`OlapException` - if error occurs

lookupMembers

```
public java.util.List lookupMembers(java.util.Set treeOps,
         java.lang.String[] nameParts)
  throws OlapException
```

(continued from last page)

Finds a collection of members in the current Cube related to a given member.

The method first looks up a member with the given fully-qualified name as for `lookupMember(String[])`, then applies the set of tree-operations to find related members.

The returned collection is sorted by level number then by member ordinal. If no member is found with the given name, the collection is empty.

For example,

```
lookupMembers(
    EnumSet.of(TreeOp.ANCESTORS, TreeOp.CHILDREN),
    "Time", "1997", "Q2")
```

returns

```
[Time].[1997]
 [Time].[1997].[Q2].[4]
 [Time].[1997].[Q2].[5]
 [Time].[1997].[Q2].[6]
```

The fully-qualified name starts with the name of the dimension, followed by the name of a root member, and continues with the name of each successive member on the path from the root member. If a member's name is unique within its level, preceding member name can be omitted.

For example, `lookupMember("Product", "Food")` and `lookupMember("Product", "All Products", "Food")` are both valid ways to locate the "Food" member of the "Product" dimension.

Parameters:

`nameParts` - Components of the fully-qualified member name

`treeOps` - Collection of tree operations to travel relative to given member in order to create list of members

Returns:

collection of members related to the given member, or empty set if the member is not found

Throws:

`OlapException` - if error occurs

org.olap4j.metadata
Class Datatype

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.metadata.Datatype
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

public final class **Datatype**
extends java.lang.Enum

Enumeration of the allowable data types of a Property or Measure.

The values derive from the OLE DB specification, specifically a subset of the OLE DB Types Indicators returned by SQL Server.

Field Summary

`public static final`	`BOOLEAN`
`public static final`	`CURRENCY`
`public static final`	`DOUBLE`
`public static final`	`INTEGER`
`public static final`	`LARGE_INTEGER`
`public static final`	`STRING`
`public static final`	`UNSIGNED_INTEGER` Used by SQL Server for colors, font flags and cell ordinal.
`public static final`	`UNSIGNED_SHORT` Used by SQL Server for font size.
`public static final`	`VARIANT` Used by SQL Server for value.

Method Summary

`static Datatype`	`forXmlaOrdinal(int xmlaOrdinal)` Looks up a Datatype by its XMLA ordinal.
`static Datatype`	`valueOf(java.lang.String name)`
`static Datatype[]`	`values()`

Methods inherited from class `java.lang.Enum` `compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `java.lang.Comparable`

`compareTo`

Fields

INTEGER

`public static final org.olap4j.metadata.Datatype` **`INTEGER`**

DOUBLE

`public static final org.olap4j.metadata.Datatype` **`DOUBLE`**

CURRENCY

`public static final org.olap4j.metadata.Datatype` **`CURRENCY`**

BOOLEAN

`public static final org.olap4j.metadata.Datatype` **`BOOLEAN`**

VARIANT

`public static final org.olap4j.metadata.Datatype` **`VARIANT`**

Used by SQL Server for value.

UNSIGNED_SHORT

`public static final org.olap4j.metadata.Datatype` **`UNSIGNED_SHORT`**

Used by SQL Server for font size.

UNSIGNED_INTEGER

`public static final org.olap4j.metadata.Datatype` **`UNSIGNED_INTEGER`**

Used by SQL Server for colors, font flags and cell ordinal.

(continued from last page)

LARGE_INTEGER

```
public static final org.olap4j.metadata.Datatype LARGE_INTEGER
```

STRING

```
public static final org.olap4j.metadata.Datatype STRING
```

Methods

values

```
public final static Datatype[] values()
```

valueOf

```
public static Datatype valueOf(java.lang.String name)
```

forXmlaOrdinal

```
public static Datatype forXmlaOrdinal(int xmlaOrdinal)
```

Looks up a Datatype by its XMLA ordinal.

Parameters:

`xmlaOrdinal` - Ordinal of a Datatype according to the XMLA specification.

Returns:

Datatype with the given ordinal, or null if there is no such Datatype

org.olap4j.metadata
Interface Dimension

All Superinterfaces:
MetadataElement

public interface **Dimension**
extends MetadataElement

An organized hierarchy of categories, known as levels, that describes data in a cube.

A Dimension typically describes a similar set of members upon which the user wants to base an analysis.

A Dimension must have at least one Hierarchy, and may have more than one, but most have exactly one Hierarchy.

Nested Class Summary

`class`	`Dimension.Type` Dimension.Type

Method Summary

`Hierarchy`	`getDefaultHierarchy()` Returns the default `Hierarchy` of this Dimension.
`Dimension.Type`	`getDimensionType()` Returns the type of this Dimension.
`NamedList`	`getHierarchies()` Returns the hierarchies in this Dimension.

Methods inherited from interface `org.olap4j.metadata.MetadataElement`

`getCaption`, `getDescription`, `getName`, `getUniqueName`

Methods

getHierarchies

`public NamedList getHierarchies()`

Returns the hierarchies in this Dimension.

Many dimensions have only one Hierarchy, whose name is the same as the Dimension.

The caller should assume that the list is immutable; if the caller modifies the list, behavior is undefined.

Returns:
hierarchies in this dimension

See Also:
`OlapDatabaseMetaData.getHierarchies(String, String, String, String, String)`

getDimensionType

```
public Dimension.Type getDimensionType()
  throws OlapException
```

Returns the type of this Dimension.

Returns:
dimension type

Throws:
`OlapException` - if database error occurs

getDefaultHierarchy

```
public Hierarchy getDefaultHierarchy()
```

Returns the default `Hierarchy` of this Dimension.

Returns:
default hierarchy

org.olap4j.metadata
Class Dimension.Type

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.metadata.Dimension.Type
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

public static final class **Dimension.Type**
extends java.lang.Enum

Enumeration of the types of a `Dimension`.

Some of the values are as specified by XMLA. For example, XMLA specifies MD_DIMTYPE_PRODUCTS with ordinal 8, which corresponds to the value `PRODUCTS`, whose `xmlaOrdinal` is 8.

See Also:
`Level$Type`, `Member$Type`, `getDimensionType`

Field Summary

`public static final`	`ACCOUNTS`
`public static final`	`BILL_OF_MATERIALS`
`public static final`	`CHANNEL`
`public static final`	`CURRENCY`
`public static final`	`CUSTOMERS`
`public static final`	`GEOGRAPHY`
`public static final`	`MEASURE` Indicates that a dimension is the Measures dimension.
`public static final`	`ORGANIZATION`
`public static final`	`OTHER`
`public static final`	`PRODUCTS`
`public static final`	`PROMOTION`
`public static final`	`QUANTITATIVE`
`public static final`	`RATES`

public static final	SCENARIO
public static final	TIME Indicates that a dimension is a time dimension.
public static final	UNKNOWN Indicates that the dimension is not related to time.
public static final	UTILITY

Method Summary

static Dimension.Type	forXmlaOrdinal(int xmlaOrdinal) Returns the type whose XMLA ordinal code is as given.
static Dimension.Type	valueOf(java.lang.String name)
static Dimension.Type[]	values()
int	xmlaOrdinal() Returns the ordinal code as specified by XMLA.

Methods inherited from class `java.lang.Enum`

`compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `java.lang.Comparable`

`compareTo`

Fields

UNKNOWN

`public static final org.olap4j.metadata.Dimension.Type` **`UNKNOWN`**

Indicates that the dimension is not related to time.

TIME

`public static final org.olap4j.metadata.Dimension.Type` **`TIME`**

Indicates that a dimension is a time dimension.

MEASURE

`public static final org.olap4j.metadata.Dimension.Type` **`MEASURE`**

Indicates that a dimension is the Measures dimension.

OTHER

`public static final org.olap4j.metadata.Dimension.Type` **`OTHER`**

QUANTITATIVE

`public static final org.olap4j.metadata.Dimension.Type` **`QUANTITATIVE`**

ACCOUNTS

`public static final org.olap4j.metadata.Dimension.Type` **`ACCOUNTS`**

CUSTOMERS

`public static final org.olap4j.metadata.Dimension.Type` **`CUSTOMERS`**

PRODUCTS

`public static final org.olap4j.metadata.Dimension.Type` **`PRODUCTS`**

SCENARIO

`public static final org.olap4j.metadata.Dimension.Type` **`SCENARIO`**

UTILITY

`public static final org.olap4j.metadata.Dimension.Type` **`UTILITY`**

CURRENCY

`public static final org.olap4j.metadata.Dimension.Type` **`CURRENCY`**

RATES

`public static final org.olap4j.metadata.Dimension.Type` **`RATES`**

CHANNEL

`public static final org.olap4j.metadata.Dimension.Type` **`CHANNEL`**

(continued from last page)

PROMOTION

```
public static final org.olap4j.metadata.Dimension.Type PROMOTION
```

ORGANIZATION

```
public static final org.olap4j.metadata.Dimension.Type ORGANIZATION
```

BILL_OF_MATERIALS

```
public static final org.olap4j.metadata.Dimension.Type BILL_OF_MATERIALS
```

GEOGRAPHY

```
public static final org.olap4j.metadata.Dimension.Type GEOGRAPHY
```

Methods

values

```
public final static Dimension.Type[] values()
```

valueOf

```
public static Dimension.Type valueOf(java.lang.String name)
```

xmlaOrdinal

```
public final int xmlaOrdinal()
```

Returns the ordinal code as specified by XMLA.

For example, the XMLA specification says that the ordinal of `PRODUCTS` is 8.

Returns:
ordinal code as specified by XMLA.

forXmlaOrdinal

```
public static Dimension.Type forXmlaOrdinal(int xmlaOrdinal)
```

Returns the type whose XMLA ordinal code is as given.

Parameters:

(continued from last page)

`xmlaOrdinal` - Ordinal code as specified by XMLA

Returns:

Dimension type, or null

org.olap4j.metadata
Interface Hierarchy

All Superinterfaces:
MetadataElement

public interface **Hierarchy**
extends MetadataElement

An organization of the set of `Members` in a `Dimension` and their positions relative to one another.

A Hierarchy is a collection of `Levels`, each of which is a category of similar `Members`.

A Dimension must have at least one Hierarchy, and may have more than one, but most have exactly one Hierarchy.

Method Summary

`Member`	`getDefaultMember()` Returns the default `Member` of this `Hierarchy`.
`Dimension`	`getDimension()` Returns the `Dimension` this `Hierarchy` belongs to.
`NamedList`	`getLevels()` Returns a list of the `Level` objects in this `Hierarchy`.
`NamedList`	`getRootMembers()` Returns the root member or members of this Dimension.
`boolean`	`hasAll()` Returns whether this `Hierarchy` has an 'all' member.

Methods inherited from interface `org.olap4j.metadata.MetadataElement`

`getCaption`, `getDescription`, `getName`, `getUniqueName`

Methods

getDimension

`public Dimension getDimension()`

Returns the `Dimension` this `Hierarchy` belongs to.

Returns:
dimension this hierarchy belongs to

getLevels

`public NamedList getLevels()`

(continued from last page)

Returns a list of the `Level` objects in this `Hierarchy`.

The caller should assume that the list is immutable; if the caller modifies the list, behavior is undefined.

Returns:
list of levels

See Also:
`OlapDatabaseMetaData.getLevels(String, String, String, String, String, String)`

hasAll

```
public boolean hasAll()
```

Returns whether this `Hierarchy` has an 'all' member.

Returns:
whether this hierarchy has an 'all' member

getDefaultMember

```
public Member getDefaultMember()
  throws OlapException
```

Returns the default `Member` of this `Hierarchy`.

If the hierarchy has an 'all' member, this member is often the default.

Returns:
the default member of this hierarchy

getRootMembers

```
public NamedList getRootMembers()
  throws OlapException
```

Returns the root member or members of this Dimension.

If the dimension has an 'all' member, then this will be the sole root member.

The caller should assume that the list is immutable; if the caller modifies the list, behavior is undefined.

The result is similar to that returned by `getLevels().get(0).getMembers()`; the contents will be the same, but this method returns a `NamedList` rather than a mere `java.util.List` because the members of the root level are known to have unique names.

Returns:
root members of this hierarchy

Throws:
`OlapException` - on database error

org.olap4j.metadata
Interface Level

All Superinterfaces:
MetadataElement

public interface **Level**
extends MetadataElement

Group of `Member` objects in a `Hierarchy`, all with the same attributes and at the same depth in the hierarchy.

Nested Class Summary

`class`	`Level.Type` Level.Type

Method Summary

`int`	`getCardinality()` Returns the number of members in this Level.
`int`	`getDepth()` Returns the depth of this `Level`.
`Dimension`	`getDimension()` Returns the Dimension this `Level` belongs to.
`Hierarchy`	`getHierarchy()` Returns the `Hierarchy` this `Level` belongs to.
`Level.Type`	`getLevelType()` Returns the type of this `Level`.
`java.util.List`	`getMembers()` Returns a list of Member objects which belong to this Level.
`NamedList`	`getProperties()` Returns a list of definitions for the properties available to members of this `Level`.

Methods inherited from interface `org.olap4j.metadata.MetadataElement`
`getCaption, getDescription, getName, getUniqueName`

Methods

getDepth

```
public int getDepth()
```

(continued from last page)

Returns the depth of this `Level`.

Note #1: In an access-controlled context, the first visible level of a hierarchy may not have a depth of 0.

Note #2: In a parent-child hierarchy, the depth of a member (as returned by may not be the same as the depth of its level.

Returns:
depth of this level

getHierarchy

```
public Hierarchy getHierarchy()
```

Returns the `Hierarchy` this `Level` belongs to.

Returns:
hierarchy this level belongs to

getDimension

```
public Dimension getDimension()
```

Returns the Dimension this `Level` belongs to. (Always equivalent to `getHierarchy().getDimension()`.)

Returns:
dimension this level belongs to

getLevelType

```
public Level.Type getLevelType()
```

Returns the type of this `Level`.

Returns:
level type

getProperties

```
public NamedList getProperties()
```

Returns a list of definitions for the properties available to members of this `Level`.

The caller should assume that the list is immutable; if the caller modifies the list, behavior is undefined.

Returns:
properties of this Level

See Also:
`OlapDatabaseMetaData.getProperties(String, String, String, String, String, String, String, String)`

getMembers

```
public java.util.List getMembers()
  throws OlapException
```

(continued from last page)

Returns a list of Member objects which belong to this Level.

Some levels have a very many members. In this case, calling this method may be expensive in space and/or time and is not recommended.

The members of a level do not have unique names, so unlike `getRootMembers()` and `getChildMembers()` the result type is a `java.util.List` not a `NamedList`.

Returns:
List of members in this Level

getCardinality

```
public int getCardinality()
```

Returns the number of members in this Level.

Returns:
number of members

org.olap4j.metadata
Class Level.Type

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.metadata.Level.Type
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

public static final class **Level.Type**
extends java.lang.Enum

Enumeration of the types of a `Level`.

Several of the values are defined by XMLA, sans the "MDLEVEL_TYPE_" prefix to their name. For example, `GEO_CONTINENT` corresponds to the value `MDLEVEL_TYPE_GEO_CONTINENT` for the `LEVEL_TYPE` property in the `MDSCHEMA_LEVELS` schema rowset.

Some of the values are specified by XMLA:

- MDLEVEL_TYPE_GEO_CONTINENT (0x2001)
- MDLEVEL_TYPE_GEO_REGION (0x2002)
- MDLEVEL_TYPE_GEO_COUNTRY (0x2003)
- MDLEVEL_TYPE_GEO_STATE_OR_PROVINCE (0x2004)
- MDLEVEL_TYPE_GEO_COUNTY (0x2005)
- MDLEVEL_TYPE_GEO_CITY (0x2006)
- MDLEVEL_TYPE_GEO_POSTALCODE (0x2007)
- MDLEVEL_TYPE_GEO_POINT (0x2008)
- MDLEVEL_TYPE_ORG_UNIT (0x1011)
- MDLEVEL_TYPE_BOM_RESOURCE (0x1012)
- MDLEVEL_TYPE_QUANTITATIVE (0x1013)
- MDLEVEL_TYPE_ACCOUNT (0x1014)
- MDLEVEL_TYPE_CUSTOMER (0x1021)
- MDLEVEL_TYPE_CUSTOMER_GROUP (0x1022)
- MDLEVEL_TYPE_CUSTOMER_HOUSEHOLD (0x1023)
- MDLEVEL_TYPE_PRODUCT (0x1031)
- MDLEVEL_TYPE_PRODUCT_GROUP (0x1032)
- MDLEVEL_TYPE_SCENARIO (0x1015)
- MDLEVEL_TYPE_UTILITY (0x1016)
- MDLEVEL_TYPE_PERSON (0x1041)
- MDLEVEL_TYPE_COMPANY (0x1042)
- MDLEVEL_TYPE_CURRENCY_SOURCE (0x1051)
- MDLEVEL_TYPE_CURRENCY_DESTINATION (0x1052)
- MDLEVEL_TYPE_CHANNEL (0x1061)
- MDLEVEL_TYPE_REPRESENTATIVE (0x1062)
- MDLEVEL_TYPE_PROMOTION (0x1071)

See Also:
`getLevelType`, `OlapDatabaseMetaData.getLevels(String, String, String, String, String, String)`

Field Summary

`public static final`	`ACCOUNT`
`public static final`	`BOM_RESOURCE`

public static final	CHANNEL
public static final	COMPANY
public static final	CURRENCY_DESTINATION
public static final	CURRENCY_SOURCE
public static final	CUSTOMER
public static final	CUSTOMER_GROUP
public static final	CUSTOMER_HOUSEHOLD
public static final	GEO_CITY
public static final	GEO_CONTINENT
public static final	GEO_COUNTRY
public static final	GEO_COUNTY
public static final	GEO_POINT
public static final	GEO_POSTALCODE
public static final	GEO_REGION
public static final	GEO_STATE_OR_PROVINCE
public static final	Null Indicates that a level holds the null member.
public static final	ORG_UNIT
public static final	PERSON
public static final	PRODUCT
public static final	PRODUCT_GROUP
public static final	PROMOTION
public static final	QUANTITATIVE
public static final	Regular Indicates that the level is not related to time.
public static final	REPRESENTATIVE

public static final	SCENARIO
public static final	TimeDays Indicates that a level refers to days.
public static final	TimeMonths Indicates that a level refers to months.
public static final	TimeQuarters Indicates that a level refers to quarters.
public static final	TimeWeeks Indicates that a level refers to weeks.
public static final	TimeYears Indicates that a level refers to years.
public static final	UTILITY

Method Summary

static Level.Type	forXmlaOrdinal(int xmlaOrdinal) Looks up a Type by its XMLA ordinal.
boolean	isTime() Returns whether this is a time-related level (TimeYears, TimeQuarters, TimeMonths, TimeWeeks, TimeDays).
static Level.Type	valueOf(java.lang.String name)
static Level.Type[]	values()
int	xmlaOrdinal() Returns the ordinal code as specified by XMLA.

Methods inherited from class `java.lang.Enum`
compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

Methods inherited from class `java.lang.Object`
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface `java.lang.Comparable`
compareTo

Fields

Regular

```
public static final org.olap4j.metadata.Level.Type Regular
```

Indicates that the level is not related to time.

TimeYears

`public static final org.olap4j.metadata.Level.Type` **`TimeYears`**

Indicates that a level refers to years. It must be used in a dimension whose type is `Dimension.Type.TIME`.

TimeQuarters

`public static final org.olap4j.metadata.Level.Type` **`TimeQuarters`**

Indicates that a level refers to quarters. It must be used in a dimension whose type is `Dimension.Type.TIME`.

TimeMonths

`public static final org.olap4j.metadata.Level.Type` **`TimeMonths`**

Indicates that a level refers to months. It must be used in a dimension whose type is `Dimension.Type.TIME`.

TimeWeeks

`public static final org.olap4j.metadata.Level.Type` **`TimeWeeks`**

Indicates that a level refers to weeks. It must be used in a dimension whose type is `Dimension.Type.TIME`.

TimeDays

`public static final org.olap4j.metadata.Level.Type` **`TimeDays`**

Indicates that a level refers to days. It must be used in a dimension whose type is `Dimension.Type.TIME`.

Null

`public static final org.olap4j.metadata.Level.Type` **`Null`**

Indicates that a level holds the null member.

GEO_CONTINENT

`public static final org.olap4j.metadata.Level.Type` **`GEO_CONTINENT`**

GEO_REGION

`public static final org.olap4j.metadata.Level.Type` **`GEO_REGION`**

GEO_COUNTRY

`public static final org.olap4j.metadata.Level.Type` **`GEO_COUNTRY`**

GEO_STATE_OR_PROVINCE

`public static final org.olap4j.metadata.Level.Type` **`GEO_STATE_OR_PROVINCE`**

(continued from last page)

GEO_COUNTY

`public static final org.olap4j.metadata.Level.Type` **`GEO_COUNTY`**

GEO_CITY

`public static final org.olap4j.metadata.Level.Type` **`GEO_CITY`**

GEO_POSTALCODE

`public static final org.olap4j.metadata.Level.Type` **`GEO_POSTALCODE`**

GEO_POINT

`public static final org.olap4j.metadata.Level.Type` **`GEO_POINT`**

ORG_UNIT

`public static final org.olap4j.metadata.Level.Type` **`ORG_UNIT`**

BOM_RESOURCE

`public static final org.olap4j.metadata.Level.Type` **`BOM_RESOURCE`**

QUANTITATIVE

`public static final org.olap4j.metadata.Level.Type` **`QUANTITATIVE`**

ACCOUNT

`public static final org.olap4j.metadata.Level.Type` **`ACCOUNT`**

CUSTOMER

`public static final org.olap4j.metadata.Level.Type` **`CUSTOMER`**

(continued from last page)

CUSTOMER_GROUP

`public static final org.olap4j.metadata.Level.Type **CUSTOMER_GROUP**`

CUSTOMER_HOUSEHOLD

`public static final org.olap4j.metadata.Level.Type **CUSTOMER_HOUSEHOLD**`

PRODUCT

`public static final org.olap4j.metadata.Level.Type **PRODUCT**`

PRODUCT_GROUP

`public static final org.olap4j.metadata.Level.Type **PRODUCT_GROUP**`

SCENARIO

`public static final org.olap4j.metadata.Level.Type **SCENARIO**`

UTILITY

`public static final org.olap4j.metadata.Level.Type **UTILITY**`

PERSON

`public static final org.olap4j.metadata.Level.Type **PERSON**`

COMPANY

`public static final org.olap4j.metadata.Level.Type **COMPANY**`

CURRENCY_SOURCE

`public static final org.olap4j.metadata.Level.Type **CURRENCY_SOURCE**`

CURRENCY_DESTINATION

`public static final org.olap4j.metadata.Level.Type **CURRENCY_DESTINATION**`

(continued from last page)

CHANNEL

```
public static final org.olap4j.metadata.Level.Type CHANNEL
```

REPRESENTATIVE

```
public static final org.olap4j.metadata.Level.Type REPRESENTATIVE
```

PROMOTION

```
public static final org.olap4j.metadata.Level.Type PROMOTION
```

Methods

values

```
public final static Level.Type[] values()
```

valueOf

```
public static Level.Type valueOf(java.lang.String name)
```

xmlaOrdinal

```
public int xmlaOrdinal()
```

Returns the ordinal code as specified by XMLA.

For example, the XMLA specification says that the ordinal of `CUSTOMER_HOUSEHOLD` is 0x1023.

Returns:
ordinal code as specified by XMLA.

forXmlaOrdinal

```
public static Level.Type forXmlaOrdinal(int xmlaOrdinal)
```

Looks up a Type by its XMLA ordinal.

Parameters:
`xmlaOrdinal` - Ordinal of a level Type according to XMLA specification.

Returns:
Type with the given ordinal, or null if there is no such Type

(continued from last page)

isTime

```
public boolean isTime()
```

Returns whether this is a time-related level (`TimeYears`, `TimeQuarters`, `TimeMonths`, `TimeWeeks`, `TimeDays`).

Returns:
whether this is a time-related level

org.olap4j.metadata
Interface Measure

All Superinterfaces:
Member, MetadataElement

public interface **Measure**
extends Member

Data value of primary interest to the user browsing the cube.

A `Measure` provides the value of each cell, and is usually numeric. Every measure is a member of a special dimension called "Measures".

Nested Class Summary

`class`	`Measure.Aggregator` Measure.Aggregator

Method Summary

`Measure.Aggregator`	`getAggregator()` Returns the Aggregator of this Measure.
`Datatype`	`getDatatype()` Returns the data type of this Measure.
`boolean`	`isVisible()` Returns whether this Measure is visible.

Methods inherited from interface `org.olap4j.metadata.Member`

`getAncestorMembers, getChildMemberCount, getChildMembers, getDataMember, getDepth, getDimension, getExpression, getHierarchy, getLevel, getMemberType, getOrdinal, getParentMember, getProperties, getPropertyFormattedValue, getPropertyValue, getSolveOrder, isAll, isCalculated, isCalculatedInQuery, isChildOrEqualTo, isHidden, setProperty`

Methods inherited from interface `org.olap4j.metadata.MetadataElement`

`getCaption, getDescription, getName, getUniqueName`

Methods

getAggregator

```
public Measure.Aggregator getAggregator()
```

Returns the Aggregator of this Measure.

Returns:
Aggregator

getDatatype

```
public Datatype getDatatype()
```

Returns the data type of this Measure.

Returns:
data type

isVisible

```
public boolean isVisible()
```

Returns whether this Measure is visible.

Returns:
whether this Measure is visible

org.olap4j.metadata
Class Measure.Aggregator

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.metadata.Measure.Aggregator
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

public static final class **Measure.Aggregator**
extends java.lang.Enum

Enumeration of the aggregate functions which can be used to derive a `Measure`.

The values are as specified by XMLA. For example, XMLA specifies MDMEASURE_AGGR_SUM with ordinal 1, which corresponds to the value `SUM`, whose `xmlaOrdinal` is 1.

Field Summary

`public static final`	`AVG` Identifies that the measure was derived using the AVG aggregation function.
`public static final`	`CALCULATED` Identifies that the measure was derived from a formula that was not any single function above.
`public static final`	`COUNT` Identifies that the measure was derived using the COUNT aggregation function.
`public static final`	`MAX` Identifies that the measure was derived using the MAX aggregation function.
`public static final`	`MIN` Identifies that the measure was derived using the MIN aggregation function.
`public static final`	`STD` Identifies that the measure was derived using the STDEV aggregation function.
`public static final`	`SUM` Identifies that the measure was derived using the SUM aggregation function.
`public static final`	`UNKNOWN` Identifies that the measure was derived from an unknown aggregation function or formula.
`public static final`	`VAR` Identifies that the measure was derived using the VAR aggregation function.

Method Summary

`static Measure.Aggregator`	`forXmlaOrdinal(int xmlaOrdinal)` Looks up an Aggregator by its XMLA ordinal.

static `Measure.Aggregator`	`valueOf(java.lang.String name)`
static `Measure.Aggregator[]`	`values()`
`int`	`xmlaOrdinal()` Returns the ordinal code as specified by XMLA.

Methods inherited from class `java.lang.Enum`
`compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `java.lang.Comparable`
`compareTo`

Fields

SUM

```
public static final org.olap4j.metadata.Measure.Aggregator SUM
```

Identifies that the measure was derived using the SUM aggregation function.

COUNT

```
public static final org.olap4j.metadata.Measure.Aggregator COUNT
```

Identifies that the measure was derived using the COUNT aggregation function.

MIN

```
public static final org.olap4j.metadata.Measure.Aggregator MIN
```

Identifies that the measure was derived using the MIN aggregation function.

MAX

```
public static final org.olap4j.metadata.Measure.Aggregator MAX
```

Identifies that the measure was derived using the MAX aggregation function.

AVG

```
public static final org.olap4j.metadata.Measure.Aggregator AVG
```

Identifies that the measure was derived using the AVG aggregation function.

VAR

```
public static final org.olap4j.metadata.Measure.Aggregator VAR
```

(continued from last page)

Identifies that the measure was derived using the VAR aggregation function.

STD

`public static final org.olap4j.metadata.Measure.Aggregator` **`STD`**

Identifies that the measure was derived using the STDEV aggregation function.

CALCULATED

`public static final org.olap4j.metadata.Measure.Aggregator` **`CALCULATED`**

Identifies that the measure was derived from a formula that was not any single function above.

UNKNOWN

`public static final org.olap4j.metadata.Measure.Aggregator` **`UNKNOWN`**

Identifies that the measure was derived from an unknown aggregation function or formula.

Methods

values

`public final static Measure.Aggregator[]` **`values`**`()`

valueOf

`public static Measure.Aggregator` **`valueOf`**`(java.lang.String name)`

xmlaOrdinal

`public final int` **`xmlaOrdinal`**`()`

Returns the ordinal code as specified by XMLA.

For example, the XMLA specification says that the ordinal of `CALCULATED` is 127.

Returns:
ordinal code as specified by XMLA.

forXmlaOrdinal

`public static Measure.Aggregator` **`forXmlaOrdinal`**`(int xmlaOrdinal)`

Looks up an Aggregator by its XMLA ordinal.

Parameters:
`xmlaOrdinal` - Ordinal of an Aggregator according to the XMLA specification.

Returns:
Aggregator with the given ordinal, or null if there is no such Aggregator

org.olap4j.metadata
Interface Member

All Superinterfaces:

MetadataElement

All Subinterfaces:

Measure

public interface **Member**
extends MetadataElement

`Member` is a data value in an OLAP Dimension.

Nested Class Summary

`class`	`Member.TreeOp` Member.TreeOp
`class`	`Member.Type` Member.Type

Method Summary

`java.util.List`	`getAncestorMembers()` Returns array of all members which are ancestor to `this`.
`int`	`getChildMemberCount()` Returns the number of children this Member has.
`NamedList`	`getChildMembers()` Returns the children of this Member, indexed by name.
`Member`	`getDataMember()` Returns the system-generated data member that is associated with a non-leaf member of a dimension.
`int`	`getDepth()` Returns the depth of this member.
`Dimension`	`getDimension()` Returns the Dimension of this Member.
`ParseTreeNode`	`getExpression()` Expression by which this member is derived, if it is a calculated member.
`Hierarchy`	`getHierarchy()` Returns the Hierarchy of this Member.
`Level`	`getLevel()` Returns the Level of this Member.
`Member.Type`	`getMemberType()` Returns the type of this Member.

`int`	`getOrdinal()` Returns the ordinal of the member.
`Member`	`getParentMember()` Returns the parent of this Member, or null if it has no parent.
`NamedList`	`getProperties()` Returns the definitions of the properties this member may have.
`java.lang.String`	`getPropertyFormattedValue(Property property)` Returns the formatted value of a given property.
`java.lang.Object`	`getPropertyValue(Property property)` Returns the value of a given property.
`int`	`getSolveOrder()` Returns the solve order of this member in a formula.
`boolean`	`isAll()` Returns whether this Member represents the aggregation of all members in its Dimension.
`boolean`	`isCalculated()` Returns whether this member is calculated using a formula.
`boolean`	`isCalculatedInQuery()` Returns whether this member is computed from a `WITH MEMBER` clause in an MDX query.
`boolean`	`isChildOrEqualTo(Member member)` Returns whether `member` is equal to, a child of, or a descendent of this Member.
`boolean`	`isHidden()` Returns whether this member is 'hidden', as per the rules which define a ragged hierarchy.
`void`	`setProperty(Property property, java.lang.Object value)` Sets a property of this member to a given value.

Methods inherited from interface `org.olap4j.metadata.MetadataElement`
`getCaption, getDescription, getName, getUniqueName`

Methods

getChildMembers

```
public NamedList getChildMembers()
  throws OlapException
```

Returns the children of this Member, indexed by name.

If access-control is in place, the list does not contain inaccessible children.

If the member has no children, returns an empty list: the result is never null.

The caller should assume that the list is immutable; if the caller modifies the list, behavior is undefined.

Returns:

children of this member

(continued from last page)

See Also:
`OlapDatabaseMetaData.getMembers(String, String, String, String, String, String, String, Set)`

getChildMemberCount

`public int getChildMemberCount()`

Returns the number of children this Member has.

This method has the same effect as `getChildMembers().size()`, but is typically less expensive.

Returns:
number of children

getParentMember

`public Member getParentMember()`

Returns the parent of this Member, or null if it has no parent.

Returns:
Parent member, or null if member has no parent

getLevel

`public Level getLevel()`

Returns the Level of this Member.

Never returns null.

Returns:
Level which this Member belongs to

getHierarchy

`public Hierarchy getHierarchy()`

Returns the Hierarchy of this Member.

Never returns null. Result is always the same as `getLevel().getHierarchy()`.

Returns:
Hierarchy which this Member belongs to

getDimension

`public Dimension getDimension()`

Returns the Dimension of this Member.

Never returns null. Result is always the same as `getLevel().getHierarchy().getDimension()`.

Returns:
Dimension which this Member belongs to

getMemberType

`public Member.Type getMemberType()`

(continued from last page)

Returns the type of this Member.

Never returns null.

Returns:
What kind of member this is

isAll

`public boolean isAll()`

Returns whether this Member represents the aggregation of all members in its Dimension.

An 'all' member is always the root of its Hierarchy; that is, its parent member is the null member, and `getRootMembers()` returns the 'all' member and no others. Some hierarchies do not have an 'all' member.

Returns:
whether this Member is the 'all' member of its Dimension

See Also:
`hasAll()`

isChildOrEqualTo

`public boolean isChildOrEqualTo(Member member)`

Returns whether `member` is equal to, a child of, or a descendent of this Member.

Parameters:
`member` - Member

Returns:
Whether the given Member is a descendent of this Member

isCalculated

`public boolean isCalculated()`

Returns whether this member is calculated using a formula.

Examples of calculated members include those defined using a `WITH MEMBER` clause in an MDX query (`getMemberType()` will return `FORMULA` for these), or a calculated member defined in a cube.

Returns:
Whether this Member is calculated

See Also:
`isCalculatedInQuery()`

getSolveOrder

`public int getSolveOrder()`

Returns the solve order of this member in a formula.

Returns:
solve order of this Member

(continued from last page)

getExpression

`public ParseTreeNode **getExpression**()`

Expression by which this member is derived, if it is a calculated member. If the member is not calulated, returns null.

Returns:
expression for this member

getAncestorMembers

`public java.util.List **getAncestorMembers**()`

Returns array of all members which are ancestor to `this`.

Returns:
ancestor Members

isCalculatedInQuery

`public boolean **isCalculatedInQuery**()`

Returns whether this member is computed from a `WITH MEMBER` clause in an MDX query. (Calculated members can also be calculated in a cube.)

Returns:
Whether this member is calculated in a query

See Also:
`isCalculated()`

getPropertyValue

`public java.lang.Object **getPropertyValue**(Property property)`

Returns the value of a given property.

Returns null if the property is not set.

Every member has certain system properties such as "name" and "caption" (the full list is described in the `Property.StandardMemberProperty` enumeration), as well as extra properties defined for its Level (see `getProperties()`).

Parameters:
`property` - Property

Returns:
formatted value of the given property

See Also:
`getPropertyFormattedValue(Property)`

getPropertyFormattedValue

`public java.lang.String **getPropertyFormattedValue**(Property property)`

(continued from last page)

Returns the formatted value of a given property.

Returns null if the property is not set.

Every member has certain system properties such as "name" and "caption" (the full list is described in the `Property.StandardMemberProperty` enumeration), as well as extra properties defined for its Level (see `getProperties()`).

Parameters:

`property` - Property

Returns:

formatted value of the given property

See Also:

`getPropertyValue(Property)`

setProperty

```
public void setProperty(Property property,
                        java.lang.Object value)
  throws OlapException
```

Sets a property of this member to a given value.

Every member has certain system properties such as "name" and "caption" (the full list is described in the `Property.StandardMemberProperty` enumeration), as well as extra properties defined for its Level (see `getProperties()`).

Parameters:

`property` - property

`value` - Property value

Throws:

`OlapException` - if the value not valid for this property (for example, a String value assigned to a Boolean property)

getProperties

```
public NamedList getProperties()
```

Returns the definitions of the properties this member may have.

For many providers, properties are defined against a Level, so result of this method will be identical to `member.getLevel().getProperties()`.

Returns:

properties of this Member

getOrdinal

```
public int getOrdinal()
```

Returns the ordinal of the member.

Returns:

ordinal of this Member

isHidden

```
public boolean isHidden()
```

(continued from last page)

Returns whether this member is 'hidden', as per the rules which define a ragged hierarchy.

Returns:
whether this member is a hidden member of a ragged hierarchy

getDepth

```
public int getDepth()
```

Returns the depth of this member.

In regular hierarchies, this is as the same as the level's depth, but in parent-child and ragged hierarchies the value may be different.

Returns:
depth of this Member

getDataMember

```
public Member getDataMember()
```

Returns the system-generated data member that is associated with a non-leaf member of a dimension.

Returns this member if this member is a leaf member, or if the non-leaf member does not have an associated data member.

Returns:
system-generated data member

org.olap4j.metadata
Class Member.Type

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.metadata.Member.Type
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

public static final class **Member.Type**
extends java.lang.Enum

Enumeration of types of members.

The values are as specified by XMLA, plus the additional `NULL` value not used by XMLA. For example, XMLA specifies `MDMEMBER_TYPE_REGULAR` with ordinal 1, which corresponds to value `REGULAR`.

The `FORMULA` value takes precedence over `MEASURE`. For example, if there is a formula (calculated) member on the Measures dimension, it is listed as `FORMULA`.

Field Summary

`public static final`	`ALL`
`public static final`	`FORMULA`
`public static final`	`MEASURE`
`public static final`	`NULL` Indicates that this member is its hierarchy's NULL member (such as is returned by the expression `[Gender].[All Gender].PrevMember`, for example).
`public static final`	`REGULAR`
`public static final`	`UNKNOWN`

Method Summary

`static Member.Type`	`valueOf(java.lang.String name)`
`static Member.Type[]`	`values()`

Methods inherited from class `java.lang.Enum`
`compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `java.lang.Comparable`
`compareTo`

Fields

UNKNOWN

`public static final org.olap4j.metadata.Member.Type` **`UNKNOWN`**

REGULAR

`public static final org.olap4j.metadata.Member.Type` **`REGULAR`**

ALL

`public static final org.olap4j.metadata.Member.Type` **`ALL`**

MEASURE

`public static final org.olap4j.metadata.Member.Type` **`MEASURE`**

FORMULA

`public static final org.olap4j.metadata.Member.Type` **`FORMULA`**

NULL

`public static final org.olap4j.metadata.Member.Type` **`NULL`**

Indicates that this member is its hierarchy's NULL member (such as is returned by the expression `[Gender].[All Gender].PrevMember`, for example).

Methods

values

`public final static Member.Type[]` **`values`**`()`

valueOf

`public static Member.Type` **`valueOf`**`(java.lang.String name)`

(continued from last page)

org.olap4j.metadata
Class Member.TreeOp

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.metadata.Member.TreeOp
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

public static final class **Member.TreeOp**
extends java.lang.Enum

Enumeration of tree operations which can be used when querying members.

Some of the values are as specified by XMLA. For example, XMLA specifies MDTREEOP_CHILDREN with ordinal 1, which corresponds to the value `CHILDREN`.

See Also:
`OlapDatabaseMetaData.getMembers(String, String, String, String, String, String, String, Set)`

Field Summary

`public static final`	`ANCESTORS` Tree operation which returns all of the ancestors.
`public static final`	`CHILDREN` Tree operation which returns only the immediate children.
`public static final`	`DESCENDANTS` Tree operation which returns all of the descendants.
`public static final`	`PARENT` Tree operation which returns only the immediate parent.
`public static final`	`SELF` Tree operation which returns itself in the list of returned rows.
`public static final`	`SIBLINGS` Tree operation which returns members on the same level.

Method Summary

`static Member.TreeOp`	`valueOf(java.lang.String name)`
`static Member.TreeOp[]`	`values()`
`int`	`xmlaOrdinal()` Returns the ordinal code as specified by XMLA.

Methods inherited from class `java.lang.Enum`

`compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `java.lang.Comparable`
`compareTo`

Fields

CHILDREN

`public static final org.olap4j.metadata.Member.TreeOp` **`CHILDREN`**

Tree operation which returns only the immediate children.

SIBLINGS

`public static final org.olap4j.metadata.Member.TreeOp` **`SIBLINGS`**

Tree operation which returns members on the same level.

PARENT

`public static final org.olap4j.metadata.Member.TreeOp` **`PARENT`**

Tree operation which returns only the immediate parent.

SELF

`public static final org.olap4j.metadata.Member.TreeOp` **`SELF`**

Tree operation which returns itself in the list of returned rows.

DESCENDANTS

`public static final org.olap4j.metadata.Member.TreeOp` **`DESCENDANTS`**

Tree operation which returns all of the descendants.

ANCESTORS

`public static final org.olap4j.metadata.Member.TreeOp` **`ANCESTORS`**

Tree operation which returns all of the ancestors.

Methods

values

`public final static Member.TreeOp[]` **`values`**`()`

(continued from last page)

valueOf

```
public static Member.TreeOp valueOf(java.lang.String name)
```

xmlaOrdinal

```
public int xmlaOrdinal()
```

Returns the ordinal code as specified by XMLA.

For example, the XMLA specification says that the ordinal of `ANCESTORS` is 32.

Returns:

ordinal code as specified by XMLA.

org.olap4j.metadata
Interface MetadataElement

All Subinterfaces:
Cube, Dimension, Hierarchy, Level, Member, Measure, NamedSet, Property

public interface **MetadataElement**
extends

An element which describes the structure of an OLAP schema.

Method Summary

`java.lang.String`	`getCaption(java.util.Locale locale)` Returns the caption of this element in the given locale.
`java.lang.String`	`getDescription(java.util.Locale locale)` Returns the description of this element in the given locale.
`java.lang.String`	`getName()` Returns the name of this element.
`java.lang.String`	`getUniqueName()` Returns the unique name of this element within its schema.

Methods

getName

```
public java.lang.String getName()
```

Returns the name of this element.

Returns:
name

getUniqueName

```
public java.lang.String getUniqueName()
```

Returns the unique name of this element within its schema.

Returns:
unique name of this element

getCaption

```
public java.lang.String getCaption(java.util.Locale locale)
```

(continued from last page)

Returns the caption of this element in the given locale.

If `locale` is null or if no caption has been defined for the element in that locale, returns the caption in base locale.

This method may return the empty string, but never returns null.

Parameters:
`locale` - Locale

Returns:
Caption of this element in the given locale, or the base locale; never null.

getDescription

`public java.lang.String **getDescription**(java.util.Locale locale)`

Returns the description of this element in the given locale.

If `locale` is null or if no description has been defined for the element in that locale, returns the description in base locale.

This method may return the empty string, but never returns null.

Parameters:
`locale` - Locale

Returns:
description of this element in the given locale, or the base locale; never null.

org.olap4j.metadata
Interface NamedList

public interface **NamedList**
extends java.util.List

Extension to `java.util.List` which allows access to members of the list by name as well as by ordinal.

Method Summary

`java.lang.Object`	`get(java.lang.String name)` Retrieves a member by name.
`int`	`indexOfName(java.lang.String name)` Returns the position where a member of a given name is found, or -1 if the member is not present.

Methods inherited from interface `java.util.List`

`add, add, addAll, addAll, clear, contains, containsAll, equals, get, hashCode, indexOf, isEmpty, iterator, lastIndexOf, listIterator, listIterator, remove, remove, removeAll, retainAll, set, size, subList, toArray, toArray`

Methods inherited from interface `java.util.Collection`

`add, addAll, clear, contains, containsAll, equals, hashCode, isEmpty, iterator, remove, removeAll, retainAll, size, toArray, toArray`

Methods inherited from interface `java.lang.Iterable`

`iterator`

Methods

get

`public java.lang.Object **get**(java.lang.String name)`

Retrieves a member by name.

Parameters:
`name` - name of the element to return

Returns:
the element of the list with the specified name, or null if there is no such element

See Also:
`List.get(int)`

indexOfName

`public int **indexOfName**(java.lang.String name)`

(continued from last page)

Returns the position where a member of a given name is found, or `-1` if the member is not present.

Parameters:

`name` - name of the element to return

Returns:

the index of element of the list with the specified name, or `-1` if there is no such element

See Also:

`List.indexOf(java.lang.Object)`

org.olap4j.metadata
Interface NamedSet

All Superinterfaces:
MetadataElement

public interface **NamedSet**
extends MetadataElement

Metadata object describing a named set defined against a `Cube`.

Method Summary

`Cube`	`getCube()` Returns the `Cube` that this `NamedSet` belongs to.
`ParseTreeNode`	`getExpression()` Returns the expression which gives the value of this NamedSet.

Methods inherited from interface `org.olap4j.metadata.MetadataElement`

`getCaption`, `getDescription`, `getName`, `getUniqueName`

Methods

getCube

```
public Cube getCube()
```

Returns the `Cube` that this `NamedSet` belongs to.

Returns:
cube this named set belongs to

getExpression

```
public ParseTreeNode getExpression()
```

Returns the expression which gives the value of this NamedSet.

Returns:
expression

org.olap4j.metadata
Interface Property

All Superinterfaces:
MetadataElement

All Known Implementing Classes:
StandardCellProperty, StandardMemberProperty

---

public interface **Property**
extends MetadataElement

Definition of a property of a `Member` or `Cell`.

Nested Class Summary

`class`	`Property.ContentType` Property.ContentType
`class`	`Property.StandardCellProperty` Property.StandardCellProperty
`class`	`Property.StandardMemberProperty` Property.StandardMemberProperty
`class`	`Property.TypeFlag` Property.TypeFlag

Method Summary

`Property.ContentType`	`getContentType()` Returns the content type of this Property.
`Datatype`	`getDatatype()` Returns the datatype of this Property.
`java.util.Set`	`getType()` Returns a set of flags which describe the type of this Property.

Methods inherited from interface `org.olap4j.metadata.MetadataElement`

`getCaption`, `getDescription`, `getName`, `getUniqueName`

Methods

getDatatype

`public Datatype getDatatype()`

Returns the datatype of this Property.

Returns:

(continued from last page)

datatype of this Property

getType

```
public java.util.Set getType()
```

Returns a set of flags which describe the type of this Property.

Returns:
type of this Property

getContentType

```
public Property.ContentType getContentType()
```

Returns the content type of this Property.

Returns:
content type

org.olap4j.metadata
Class Property.TypeFlag

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.metadata.Property.TypeFlag
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

public static final class **Property.TypeFlag**
extends java.lang.Enum

Enumeration of aspects of the type of a Property. In particular, whether it belongs to a member or a cell.

The values are as specified by XMLA for the PROPERTY_TYPE attribute of the MDSCHEMA_PROPERTIES data set. For example, XMLA specifies that the value 9 (0x1 | 0x8) means that a property belongs to a member and is a binary large object (BLOB). In this case, `getType` will return the `java.util.Set` {`MEMBER`, `BLOB`}.

Field Summary

`public static final`	`BLOB` Identifies a property which contains a binary large object (blob).
`public static final`	`CELL` Identifies a property of a cell.
`public static final`	`MEMBER` Identifies a property of a member.
`public static final`	`SYSTEM` Identifies an internal property.
`public final`	`xmlaOrdinal`

Method Summary

`static java.util.Set`	`forMask(int xmlaOrdinalMask)` Creates a set of TypeFlag values by parsing a mask.
`static Property.TypeFlag`	`forXmlaOrdinal(int xmlaOrdinal)` Looks up a TypeFlag by its XMLA ordinal.
`static Property.TypeFlag`	`valueOf(java.lang.String name)`
`static Property.TypeFlag[]`	`values()`

Methods inherited from class `java.lang.Enum`

`compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf`

Methods inherited from class `java.lang.Object` `equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `java.lang.Comparable`

`compareTo`

Fields

MEMBER

`public static final org.olap4j.metadata.Property.TypeFlag` **`MEMBER`**

Identifies a property of a member. This property can be used in the DIMENSION PROPERTIES clause of the SELECT statement.

CELL

`public static final org.olap4j.metadata.Property.TypeFlag` **`CELL`**

Identifies a property of a cell. This property can be used in the CELL PROPERTIES clause that occurs at the end of the SELECT statement.

SYSTEM

`public static final org.olap4j.metadata.Property.TypeFlag` **`SYSTEM`**

Identifies an internal property.

BLOB

`public static final org.olap4j.metadata.Property.TypeFlag` **`BLOB`**

Identifies a property which contains a binary large object (blob).

xmlaOrdinal

`public final int` **`xmlaOrdinal`**

Methods

values

`public final static Property.TypeFlag[]` **`values`**`()`

valueOf

`public static Property.TypeFlag` **`valueOf`**`(java.lang.String name)`

(continued from last page)

forXmlaOrdinal

```
public static Property.TypeFlag forXmlaOrdinal(int xmlaOrdinal)
```

Looks up a TypeFlag by its XMLA ordinal.

Parameters:

`xmlaOrdinal` - Ordinal of a TypeFlag according to the XMLA specification.

Returns:

TypeFlag with the given ordinal, or null if there is no such TypeFlag

forMask

```
public static java.util.Set forMask(int xmlaOrdinalMask)
```

Creates a set of TypeFlag values by parsing a mask.

For example, `forMask(9)` returns the set {`MEMBER`, `BLOB`} because 9 = MEMBER (1) | BLOB (8).

Parameters:

`xmlaOrdinalMask` - Bit mask

Returns:

Set of TypeFlag values

org.olap4j.metadata
Class Property.StandardMemberProperty

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.metadata.Property.StandardMemberProperty
```

All Implemented Interfaces:
Property, java.io.Serializable, java.lang.Comparable

public static final class **Property.StandardMemberProperty**
extends java.lang.Enum
implements java.lang.Comparable, java.io.Serializable, Property

Enumeration of the system properties available for every `Member`.

The following properties are mandatory for members:

- `CATALOG_NAME`
- `SCHEMA_NAME`
- `CUBE_NAME`
- `DIMENSION_UNIQUE_NAME`
- `HIERARCHY_UNIQUE_NAME`
- `LEVEL_UNIQUE_NAME`
- `LEVEL_NUMBER`
- `MEMBER_UNIQUE_NAME`
- `MEMBER_NAME`
- `MEMBER_TYPE`
- `MEMBER_GUID`
- `MEMBER_CAPTION`
- `MEMBER_ORDINAL`
- `CHILDREN_CARDINALITY`
- `PARENT_LEVEL`
- `PARENT_UNIQUE_NAME`
- `PARENT_COUNT`
- `DESCRIPTION`

Field Summary

`public static final`	`$visible` Definition of the internal property which holds the name of the system property which determines whether to show a member (especially a measure or calculated member) in a user interface such as JPivot.
`public static final`	`CATALOG_NAME` Definition of the property which holds the name of the current catalog.
`public static final`	`CHILDREN_CARDINALITY` Definition of the property which holds the number of children this member has.
`public static final`	`CUBE_NAME` Definition of the property which holds the name of the current cube.
`public static final`	`DEPTH` Definition of the property which holds the level depth of a member.

public static final	DESCRIPTION Definition of the property which holds the description of this member.
public static final	DIMENSION_UNIQUE_NAME Definition of the property which holds the unique name of the current dimension.
public static final	DISPLAY_INFO Definition of the property which holds the DISPLAY_INFO required by XML/A.
public static final	HIERARCHY_UNIQUE_NAME Definition of the property which holds the unique name of the current hierarchy.
public static final	IS_DATAMEMBER Definition of the property that indicates whether the member is a data member.
public static final	IS_PLACEHOLDERMEMBER Definition of the boolean property that indicates whether a member is a placeholder member for an empty position in a dimension hierarchy.
public static final	LEVEL_NUMBER Definition of the property which holds the ordinal of the current level.
public static final	LEVEL_UNIQUE_NAME Definition of the property which holds the unique name of the current level.
public static final	MEMBER_CAPTION Definition of the property which holds the label or caption associated with the member, or the member's name if no caption is defined.
public static final	MEMBER_GUID Definition of the property which holds the GUID of the member
public static final	MEMBER_KEY Definition of the internal property which holds the value of the member key in the original data type.
public static final	MEMBER_NAME Definition of the property which holds the name of the current member.
public static final	MEMBER_ORDINAL Definition of the property which holds the ordinal of the current member.
public static final	MEMBER_TYPE Definition of the property which holds the type of the member.
public static final	MEMBER_UNIQUE_NAME Definition of the property which holds the unique name of the current member.
public static final	PARENT_COUNT Definition of the property which holds the number of parents that this member has.
public static final	PARENT_LEVEL Definition of the property which holds the distance from the root of the hierarchy of this member's parent.
public static final	PARENT_UNIQUE_NAME Definition of the property which holds the Name of the current catalog.
public static final	SCHEMA_NAME Definition of the property which holds the name of the current schema.

public static final	VALUE Definition of the property which holds the value of a cell.

Method Summary

java.lang.String	getCaption(java.util.Locale locale)
Property.ContentType	getContentType()
Datatype	getDatatype()
java.lang.String	getDescription(java.util.Locale locale)
java.lang.String	getName()
java.util.Set	getType()
java.lang.String	getUniqueName()
boolean	isInternal()
static Property.StandardMemberProperty	valueOf(java.lang.String name)
static Property.StandardMemberProperty[]	values()

Methods inherited from class `java.lang.Enum`

`compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `java.lang.Comparable`

`compareTo`

Methods inherited from interface `org.olap4j.metadata.Property`

`getContentType, getDatatype, getType`

Methods inherited from interface `org.olap4j.metadata.MetadataElement`

`getCaption, getDescription, getName, getUniqueName`

Fields

(continued from last page)

CATALOG_NAME

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`CATALOG_NAME`**

Definition of the property which holds the name of the current catalog.

SCHEMA_NAME

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`SCHEMA_NAME`**

Definition of the property which holds the name of the current schema.

CUBE_NAME

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`CUBE_NAME`**

Definition of the property which holds the name of the current cube.

DIMENSION_UNIQUE_NAME

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`DIMENSION_UNIQUE_NAME`**

Definition of the property which holds the unique name of the current dimension.

HIERARCHY_UNIQUE_NAME

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`HIERARCHY_UNIQUE_NAME`**

Definition of the property which holds the unique name of the current hierarchy.

LEVEL_UNIQUE_NAME

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`LEVEL_UNIQUE_NAME`**

Definition of the property which holds the unique name of the current level.

LEVEL_NUMBER

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`LEVEL_NUMBER`**

Definition of the property which holds the ordinal of the current level.

MEMBER_ORDINAL

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`MEMBER_ORDINAL`**

Definition of the property which holds the ordinal of the current member.

MEMBER_NAME

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`MEMBER_NAME`**

Definition of the property which holds the name of the current member.

(continued from last page)

MEMBER_UNIQUE_NAME

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`MEMBER_UNIQUE_NAME`**

Definition of the property which holds the unique name of the current member.

MEMBER_TYPE

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`MEMBER_TYPE`**

Definition of the property which holds the type of the member.

MEMBER_GUID

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`MEMBER_GUID`**

Definition of the property which holds the GUID of the member

MEMBER_CAPTION

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`MEMBER_CAPTION`**

Definition of the property which holds the label or caption associated with the member, or the member's name if no caption is defined.

CHILDREN_CARDINALITY

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`CHILDREN_CARDINALITY`**

Definition of the property which holds the number of children this member has.

PARENT_LEVEL

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`PARENT_LEVEL`**

Definition of the property which holds the distance from the root of the hierarchy of this member's parent.

PARENT_UNIQUE_NAME

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`PARENT_UNIQUE_NAME`**

Definition of the property which holds the Name of the current catalog.

PARENT_COUNT

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`PARENT_COUNT`**

Definition of the property which holds the number of parents that this member has. Generally 1, or 0 for root members.

DESCRIPTION

`public static final org.olap4j.metadata.Property.StandardMemberProperty` **`DESCRIPTION`**

Definition of the property which holds the description of this member.

(continued from last page)

$visible

```
public static final org.olap4j.metadata.Property.StandardMemberProperty $visible
```

Definition of the internal property which holds the name of the system property which determines whether to show a member (especially a measure or calculated member) in a user interface such as JPivot.

MEMBER_KEY

```
public static final org.olap4j.metadata.Property.StandardMemberProperty MEMBER_KEY
```

Definition of the internal property which holds the value of the member key in the original data type. MEMBER_KEY is for backward-compatibility. MEMBER_KEY has the same value as KEY0 for non-composite keys, and MEMBER_KEY property is null for composite keys.

IS_PLACEHOLDERMEMBER

```
public static final org.olap4j.metadata.Property.StandardMemberProperty
IS_PLACEHOLDERMEMBER
```

Definition of the boolean property that indicates whether a member is a placeholder member for an empty position in a dimension hierarchy.

IS_DATAMEMBER

```
public static final org.olap4j.metadata.Property.StandardMemberProperty IS_DATAMEMBER
```

Definition of the property that indicates whether the member is a data member.

DEPTH

```
public static final org.olap4j.metadata.Property.StandardMemberProperty DEPTH
```

Definition of the property which holds the level depth of a member.

Caution: Level depth of members in parent-child hierarchy isn't from their levels. It's calculated from the underlying data dynamically.

DISPLAY_INFO

```
public static final org.olap4j.metadata.Property.StandardMemberProperty DISPLAY_INFO
```

Definition of the property which holds the DISPLAY_INFO required by XML/A.

Caution: This property's value is calculated based on a specified MDX query, so its value is dynamic at runtime.

VALUE

```
public static final org.olap4j.metadata.Property.StandardMemberProperty VALUE
```

Definition of the property which holds the value of a cell. Is usually numeric (since most measures are numeric) but is occasionally another type.

Methods

values

```
public final static Property.StandardMemberProperty[] values()
```

valueOf

`public static Property.StandardMemberProperty **valueOf**(java.lang.String name)`

getName

`public java.lang.String **getName**()`

getUniqueName

`public java.lang.String **getUniqueName**()`

getDescription

`public java.lang.String **getDescription**(java.util.Locale locale)`

getCaption

`public java.lang.String **getCaption**(java.util.Locale locale)`

getDatatype

`public Datatype **getDatatype**()`

getType

`public java.util.Set **getType**()`

getContentType

`public Property.ContentType **getContentType**()`

isInternal

`public boolean **isInternal**()`

org.olap4j.metadata
Class Property.StandardCellProperty

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.metadata.Property.StandardCellProperty
```

All Implemented Interfaces:
Property, java.io.Serializable, java.lang.Comparable

public static final class **Property.StandardCellProperty**
extends java.lang.Enum
implements java.lang.Comparable, java.io.Serializable, Property

Enumeration of the system properties available for every `Cell`.

The following propertiess are mandatory for cells:

- `BACK_COLOR`
- `CELL_EVALUATION_LIST`
- `CELL_ORDINAL`
- `FORE_COLOR`
- `FONT_NAME`
- `FONT_SIZE`
- `FONT_FLAGS`
- `FORMAT_STRING`
- `FORMATTED_VALUE`
- `NON_EMPTY_BEHAVIOR`
- `SOLVE_ORDER`
- `VALUE`

Field Summary

`public static final`	`BACK_COLOR`
`public static final`	`CELL_EVALUATION_LIST`
`public static final`	`CELL_ORDINAL`
`public static final`	`DATATYPE` Definition of the property which holds the datatype of a cell.
`public static final`	`FONT_FLAGS`
`public static final`	`FONT_NAME`
`public static final`	`FONT_SIZE`
`public static final`	`FORE_COLOR`

public static final	FORMAT_STRING Definition of the property which holds the format string used to format cell values.
public static final	FORMATTED_VALUE Definition of the property which holds the formatted value of a cell.
public static final	NON_EMPTY_BEHAVIOR
public static final	SOLVE_ORDER Definition of the property which determines the solve order of a calculated member with respect to other calculated members.
public static final	VALUE Definition of the property which holds the value of a cell.

Method Summary

java.lang.String	getCaption(java.util.Locale locale)
Property.ContentType	getContentType()
Datatype	getDatatype()
java.lang.String	getDescription(java.util.Locale locale)
java.lang.String	getName()
java.util.Set	getType()
java.lang.String	getUniqueName()
boolean	isInternal()
static Property.StandardCellProperty	valueOf(java.lang.String name)
static Property.StandardCellProperty[]	values()

Methods inherited from class java.lang.Enum

compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface java.lang.Comparable

compareTo

Methods inherited from interface org.olap4j.metadata.Property getContentType, getDatatype, getType

Methods inherited from interface org.olap4j.metadata.MetadataElement

getCaption, getDescription, getName, getUniqueName

Fields

BACK_COLOR

`public static final org.olap4j.metadata.Property.StandardCellProperty` **`BACK_COLOR`**

CELL_EVALUATION_LIST

`public static final org.olap4j.metadata.Property.StandardCellProperty` **`CELL_EVALUATION_LIST`**

CELL_ORDINAL

`public static final org.olap4j.metadata.Property.StandardCellProperty` **`CELL_ORDINAL`**

FORE_COLOR

`public static final org.olap4j.metadata.Property.StandardCellProperty` **`FORE_COLOR`**

FONT_NAME

`public static final org.olap4j.metadata.Property.StandardCellProperty` **`FONT_NAME`**

FONT_SIZE

`public static final org.olap4j.metadata.Property.StandardCellProperty` **`FONT_SIZE`**

FONT_FLAGS

`public static final org.olap4j.metadata.Property.StandardCellProperty` **`FONT_FLAGS`**

FORMATTED_VALUE

`public static final org.olap4j.metadata.Property.StandardCellProperty` **`FORMATTED_VALUE`**

(continued from last page)

Definition of the property which holds the formatted value of a cell.

FORMAT_STRING

```
public static final org.olap4j.metadata.Property.StandardCellProperty FORMAT_STRING
```

Definition of the property which holds the format string used to format cell values.

NON_EMPTY_BEHAVIOR

```
public static final org.olap4j.metadata.Property.StandardCellProperty
NON_EMPTY_BEHAVIOR
```

SOLVE_ORDER

```
public static final org.olap4j.metadata.Property.StandardCellProperty SOLVE_ORDER
```

Definition of the property which determines the solve order of a calculated member with respect to other calculated members.

VALUE

```
public static final org.olap4j.metadata.Property.StandardCellProperty VALUE
```

Definition of the property which holds the value of a cell. Is usually numeric (since most measures are numeric) but is occasionally another type.

DATATYPE

```
public static final org.olap4j.metadata.Property.StandardCellProperty DATATYPE
```

Definition of the property which holds the datatype of a cell. Valid values are "String", "Numeric", "Integer". The property's value derives from the "datatype" attribute of the "Measure" element; if the datatype attribute is not specified, the datatype is "Numeric" by default, except measures whose aggregator is "Count", whose datatype is "Integer".

Methods

values

```
public final static Property.StandardCellProperty[] values()
```

valueOf

```
public static Property.StandardCellProperty valueOf(java.lang.String name)
```

getDatatype

```
public Datatype getDatatype()
```

(continued from last page)

getType

```
public java.util.Set getType()
```

getName

```
public java.lang.String getName()
```

getUniqueName

```
public java.lang.String getUniqueName()
```

getCaption

```
public java.lang.String getCaption(java.util.Locale locale)
```

getDescription

```
public java.lang.String getDescription(java.util.Locale locale)
```

isInternal

```
public boolean isInternal()
```

getContentType

```
public Property.ContentType getContentType()
```

org.olap4j.metadata
Class Property.ContentType

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.metadata.Property.ContentType
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

public static final class **Property.ContentType**
extends java.lang.Enum

Enumeration of the types of a `Property`.

The values are as specified by XMLA. For example, XMLA specifies MD_PROPTYPE_CAPTION with ordinal 0x21, which corresponds to the value `CAPTION`, whose `xmlaOrdinal` is 0x21.

Field Summary

`public static final`	`ADDRESS`
`public static final`	`ADDRESS_BUILDING`
`public static final`	`ADDRESS_CITY`
`public static final`	`ADDRESS_COUNTRY`
`public static final`	`ADDRESS_FAX`
`public static final`	`ADDRESS_FLOOR`
`public static final`	`ADDRESS_HOUSE`
`public static final`	`ADDRESS_PHONE`
`public static final`	`ADDRESS_QUARTER`
`public static final`	`ADDRESS_ROOM`
`public static final`	`ADDRESS_STATE_OR_PROVINCE`
`public static final`	`ADDRESS_STREET`
`public static final`	`ADDRESS_ZIP`
`public static final`	`CAPTION`

public static final	CAPTION_ABREVIATION
public static final	CAPTION_DESCRIPTION
public static final	CAPTION_SHORT
public static final	DATE
public static final	DATE_CANCELED
public static final	DATE_DURATION
public static final	DATE_ENDED
public static final	DATE_MODIFIED
public static final	DATE_START
public static final	FORMATTING_COLOR
public static final	FORMATTING_FONT
public static final	FORMATTING_FONT_EFFECTS
public static final	FORMATTING_FONT_SIZE
public static final	FORMATTING_ORDER
public static final	FORMATTING_SUB_TOTAL
public static final	GEO_BOUNDARY_BOTTOM
public static final	GEO_BOUNDARY_FRONT
public static final	GEO_BOUNDARY_LEFT
public static final	GEO_BOUNDARY_POLYGON
public static final	GEO_BOUNDARY_REAR
public static final	GEO_BOUNDARY_RIGHT
public static final	GEO_BOUNDARY_TOP
public static final	GEO_CENTROID_X
public static final	GEO_CENTROID_Y

public static final	GEO_CENTROID_Z
public static final	ID
public static final	ORG_TITLE
public static final	PERSON_CONTACT
public static final	PERSON_DEMOGRAPHIC
public static final	PERSON_FIRST_NAME
public static final	PERSON_FULL_NAME
public static final	PERSON_LAST_NAME
public static final	PERSON_MIDDLE_NAME
public static final	PHYSICAL_COLOR
public static final	PHYSICAL_DENSITY
public static final	PHYSICAL_DEPTH
public static final	PHYSICAL_HEIGHT
public static final	PHYSICAL_SIZE
public static final	PHYSICAL_VOLUME
public static final	PHYSICAL_WEIGHT
public static final	PHYSICAL_WIDTH
public static final	QTY_RANGE_HIGH
public static final	QTY_RANGE_LOW
public static final	REGULAR
public static final	RELATION_TO_PARENT
public static final	ROLLUP_OPERATOR
public static final	VERSION
public static final	WEB_HTML

public static final	WEB_MAIL_ALIAS
public static final	WEB_URL
public static final	WEB_XML_OR_XSL

Method Summary

static Property.ContentType	forXmlaOrdinal(int xmlaOrdinal) Looks up a ContentType by its XMLA ordinal.
static Property.ContentType	valueOf(java.lang.String name)
static Property.ContentType[]	values()
int	xmlaOrdinal() Returns the ordinal code as specified by XMLA.

Methods inherited from class `java.lang.Enum`
compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

Methods inherited from class `java.lang.Object`
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface `java.lang.Comparable`
compareTo

Fields

REGULAR

`public static final org.olap4j.metadata.Property.ContentType` **`REGULAR`**

ID

`public static final org.olap4j.metadata.Property.ContentType` **`ID`**

RELATION_TO_PARENT

`public static final org.olap4j.metadata.Property.ContentType` **`RELATION_TO_PARENT`**

(continued from last page)

ROLLUP_OPERATOR

```
public static final org.olap4j.metadata.Property.ContentType ROLLUP_OPERATOR
```

ORG_TITLE

```
public static final org.olap4j.metadata.Property.ContentType ORG_TITLE
```

CAPTION

```
public static final org.olap4j.metadata.Property.ContentType CAPTION
```

CAPTION_SHORT

```
public static final org.olap4j.metadata.Property.ContentType CAPTION_SHORT
```

CAPTION_DESCRIPTION

```
public static final org.olap4j.metadata.Property.ContentType CAPTION_DESCRIPTION
```

CAPTION_ABREVIATION

```
public static final org.olap4j.metadata.Property.ContentType CAPTION_ABREVIATION
```

WEB_URL

```
public static final org.olap4j.metadata.Property.ContentType WEB_URL
```

WEB_HTML

```
public static final org.olap4j.metadata.Property.ContentType WEB_HTML
```

WEB_XML_OR_XSL

```
public static final org.olap4j.metadata.Property.ContentType WEB_XML_OR_XSL
```

WEB_MAIL_ALIAS

```
public static final org.olap4j.metadata.Property.ContentType WEB_MAIL_ALIAS
```

(continued from last page)

ADDRESS

`public static final org.olap4j.metadata.Property.ContentType` **`ADDRESS`**

ADDRESS_STREET

`public static final org.olap4j.metadata.Property.ContentType` **`ADDRESS_STREET`**

ADDRESS_HOUSE

`public static final org.olap4j.metadata.Property.ContentType` **`ADDRESS_HOUSE`**

ADDRESS_CITY

`public static final org.olap4j.metadata.Property.ContentType` **`ADDRESS_CITY`**

ADDRESS_STATE_OR_PROVINCE

`public static final org.olap4j.metadata.Property.ContentType` **`ADDRESS_STATE_OR_PROVINCE`**

ADDRESS_ZIP

`public static final org.olap4j.metadata.Property.ContentType` **`ADDRESS_ZIP`**

ADDRESS_QUARTER

`public static final org.olap4j.metadata.Property.ContentType` **`ADDRESS_QUARTER`**

ADDRESS_COUNTRY

`public static final org.olap4j.metadata.Property.ContentType` **`ADDRESS_COUNTRY`**

ADDRESS_BUILDING

`public static final org.olap4j.metadata.Property.ContentType` **`ADDRESS_BUILDING`**

(continued from last page)

ADDRESS_ROOM

`public static final org.olap4j.metadata.Property.ContentType` **`ADDRESS_ROOM`**

ADDRESS_FLOOR

`public static final org.olap4j.metadata.Property.ContentType` **`ADDRESS_FLOOR`**

ADDRESS_FAX

`public static final org.olap4j.metadata.Property.ContentType` **`ADDRESS_FAX`**

ADDRESS_PHONE

`public static final org.olap4j.metadata.Property.ContentType` **`ADDRESS_PHONE`**

GEO_CENTROID_X

`public static final org.olap4j.metadata.Property.ContentType` **`GEO_CENTROID_X`**

GEO_CENTROID_Y

`public static final org.olap4j.metadata.Property.ContentType` **`GEO_CENTROID_Y`**

GEO_CENTROID_Z

`public static final org.olap4j.metadata.Property.ContentType` **`GEO_CENTROID_Z`**

GEO_BOUNDARY_TOP

`public static final org.olap4j.metadata.Property.ContentType` **`GEO_BOUNDARY_TOP`**

GEO_BOUNDARY_LEFT

`public static final org.olap4j.metadata.Property.ContentType` **`GEO_BOUNDARY_LEFT`**

GEO_BOUNDARY_BOTTOM

`public static final org.olap4j.metadata.Property.ContentType` **`GEO_BOUNDARY_BOTTOM`**

(continued from last page)

GEO_BOUNDARY_RIGHT

```
public static final org.olap4j.metadata.Property.ContentType GEO_BOUNDARY_RIGHT
```

GEO_BOUNDARY_FRONT

```
public static final org.olap4j.metadata.Property.ContentType GEO_BOUNDARY_FRONT
```

GEO_BOUNDARY_REAR

```
public static final org.olap4j.metadata.Property.ContentType GEO_BOUNDARY_REAR
```

GEO_BOUNDARY_POLYGON

```
public static final org.olap4j.metadata.Property.ContentType GEO_BOUNDARY_POLYGON
```

PHYSICAL_SIZE

```
public static final org.olap4j.metadata.Property.ContentType PHYSICAL_SIZE
```

PHYSICAL_COLOR

```
public static final org.olap4j.metadata.Property.ContentType PHYSICAL_COLOR
```

PHYSICAL_WEIGHT

```
public static final org.olap4j.metadata.Property.ContentType PHYSICAL_WEIGHT
```

PHYSICAL_HEIGHT

```
public static final org.olap4j.metadata.Property.ContentType PHYSICAL_HEIGHT
```

PHYSICAL_WIDTH

```
public static final org.olap4j.metadata.Property.ContentType PHYSICAL_WIDTH
```

(continued from last page)

PHYSICAL_DEPTH

public static final org.olap4j.metadata.Property.ContentType **PHYSICAL_DEPTH**

PHYSICAL_VOLUME

public static final org.olap4j.metadata.Property.ContentType **PHYSICAL_VOLUME**

PHYSICAL_DENSITY

public static final org.olap4j.metadata.Property.ContentType **PHYSICAL_DENSITY**

PERSON_FULL_NAME

public static final org.olap4j.metadata.Property.ContentType **PERSON_FULL_NAME**

PERSON_FIRST_NAME

public static final org.olap4j.metadata.Property.ContentType **PERSON_FIRST_NAME**

PERSON_LAST_NAME

public static final org.olap4j.metadata.Property.ContentType **PERSON_LAST_NAME**

PERSON_MIDDLE_NAME

public static final org.olap4j.metadata.Property.ContentType **PERSON_MIDDLE_NAME**

PERSON_DEMOGRAPHIC

public static final org.olap4j.metadata.Property.ContentType **PERSON_DEMOGRAPHIC**

PERSON_CONTACT

public static final org.olap4j.metadata.Property.ContentType **PERSON_CONTACT**

QTY_RANGE_LOW

public static final org.olap4j.metadata.Property.ContentType **QTY_RANGE_LOW**

(continued from last page)

QTY_RANGE_HIGH

public static final org.olap4j.metadata.Property.ContentType **QTY_RANGE_HIGH**

FORMATTING_COLOR

public static final org.olap4j.metadata.Property.ContentType **FORMATTING_COLOR**

FORMATTING_ORDER

public static final org.olap4j.metadata.Property.ContentType **FORMATTING_ORDER**

FORMATTING_FONT

public static final org.olap4j.metadata.Property.ContentType **FORMATTING_FONT**

FORMATTING_FONT_EFFECTS

public static final org.olap4j.metadata.Property.ContentType **FORMATTING_FONT_EFFECTS**

FORMATTING_FONT_SIZE

public static final org.olap4j.metadata.Property.ContentType **FORMATTING_FONT_SIZE**

FORMATTING_SUB_TOTAL

public static final org.olap4j.metadata.Property.ContentType **FORMATTING_SUB_TOTAL**

DATE

public static final org.olap4j.metadata.Property.ContentType **DATE**

DATE_START

public static final org.olap4j.metadata.Property.ContentType **DATE_START**

(continued from last page)

DATE_ENDED

`public static final org.olap4j.metadata.Property.ContentType` **`DATE_ENDED`**

DATE_CANCELED

`public static final org.olap4j.metadata.Property.ContentType` **`DATE_CANCELED`**

DATE_MODIFIED

`public static final org.olap4j.metadata.Property.ContentType` **`DATE_MODIFIED`**

DATE_DURATION

`public static final org.olap4j.metadata.Property.ContentType` **`DATE_DURATION`**

VERSION

`public static final org.olap4j.metadata.Property.ContentType` **`VERSION`**

Methods

values

`public final static Property.ContentType[]` **`values`**`()`

---

valueOf

`public static Property.ContentType` **`valueOf`**`(java.lang.String name)`

---

xmlaOrdinal

`public int` **`xmlaOrdinal`**`()`

Returns the ordinal code as specified by XMLA.

For example, the XMLA specification says that the ordinal of `FORMATTING_FONT_EFFECTS` is 0xA4.

Returns:

ordinal code as specified by XMLA.

---

forXmlaOrdinal

`public static Property.ContentType` **`forXmlaOrdinal`**`(int xmlaOrdinal)`

(continued from last page)

Looks up a ContentType by its XMLA ordinal.

Parameters:

`xmlaOrdinal` - Ordinal of a ContentType according to the XMLA specification.

Returns:

ContentType with the given ordinal, or null if there is no such ContentType

org.olap4j.metadata
Interface Schema

public interface **Schema**
extends

A collection of database objects that contain structural information, or metadata, about a database.

A Schema belongs to a `Catalog` and contains a number of `Cubes` and shared `Dimensions`.

Method Summary

`Catalog`	`getCatalog()` Returns the `Catalog` this `Schema` belongs to.
`NamedList`	`getCubes()` Returns a list of cubes in this `Schema`.
`java.lang.String`	`getName()` Returns the name of this Schema.
`NamedList`	`getSharedDimensions()` Returns a list of shared `Dimension` objects in this `Schema`.
`java.util.Collection`	`getSupportedLocales()` Returns a collection of `java.util.Locale` objects for which this `Schema` has been localized.

Methods

getCatalog

```
public Catalog getCatalog()
```

Returns the `Catalog` this `Schema` belongs to.

Returns:
catalog this schema belongs to

getName

```
public java.lang.String getName()
```

Returns the name of this Schema.

Returns:
name of this Schema

getCubes

```
public NamedList getCubes()
  throws OlapException
```

(continued from last page)

Returns a list of cubes in this `Schema`.

The caller should assume that the list is immutable; if the caller modifies the list, behavior is undefined.

Returns:
List of cubes in this Schema

Throws:
`OlapException` - if database error occurs

See Also:
`OlapDatabaseMetaData.getCubes(String, String, String)`

getSharedDimensions

```
public NamedList getSharedDimensions()
  throws OlapException
```

Returns a list of shared `Dimension` objects in this `Schema`.

The caller should assume that the list is immutable; if the caller modifies the list, behavior is undefined.

Returns:
list of shared dimensions

Throws:
`OlapException` - if database error occurs

See Also:
`OlapDatabaseMetaData.getDimensions(String, String, String, String)`

getSupportedLocales

```
public java.util.Collection getSupportedLocales()
  throws OlapException
```

Returns a collection of `java.util.Locale` objects for which this `Schema` has been localized.

Consider the following use case. Suppose one cube is available in English and French, and in French and Spanish, and both are shown in same portal. Clients typically say that seeing reports in a mixture of languages is confusing; the portal would figure out the best common language, in this case French. This method allows the client to choose the most appropriate locale.

The list is advisory: a client is free to choose another locale, in which case, the server will probably revert to the base locale for locale-specific behavior such as captions and formatting.

Returns:
List of locales for which this `Schema` has been localized

Throws:
`OlapException` - if database error occurs

See Also:
`getSupportedLocales`

Package org.olap4j.query

Provides an object model for building OLAP queries programmatically (experimental).

NOTE: This package is experimental. Classes may be renamed or removed in a future release of olap4j.

org.olap4j.query
Interface CellSetFormatter

All Known Implementing Classes:
RectangularCellSetFormatter, TraditionalCellSetFormatter

public interface **CellSetFormatter**
extends

Converts a `CellSet` into text.

This interface is experimental. It is not part of the olap4j specification and is subject to change without notice.

Method Summary

`void`	`format(CellSet cellSet, java.io.PrintWriter pw)` Formats a CellSet as text to a PrintWriter.

Methods

format

```
public void format(CellSet cellSet,
          java.io.PrintWriter pw)
```

Formats a CellSet as text to a PrintWriter.

Parameters:
`cellSet` - Cell set
`pw` - Print writer

org.olap4j.query
Class Query

```
java.lang.Object
  └-org.olap4j.query.Query
```

public class **Query**
extends java.lang.Object

Query model.

Constructor Summary

public	`Query(java.lang.String name, Cube cube)`

Method Summary

`CellSet`	`execute()` Executes the query against the current OlapConnection and returns a CellSet object representation of the data.
`java.util.Map`	`getAxes()` Returns a map of the current query's axis.
`Cube`	`getCube()` Returns the underlying cube object that is used to query against.
`QueryDimension`	`getDimension(java.lang.String name)` Returns the Olap4j's Dimension object according to the name given as a parameter.
`java.util.Locale`	`getLocale()` Returns the current locale with which this query is expressed.
`java.lang.String`	`getName()` Returns this query's name.
`SelectNode`	`getSelect()` Returns the MDX parse tree behind this Query.
`QueryAxis`	`getUnusedAxis()` Returns the fictional axis into which all unused dimensions are stored.
`void`	`swapAxes()` Swaps rows and columns axes.
`boolean`	`validate()`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

(continued from last page)

Constructors

Query

```
public Query(java.lang.String name,
             Cube cube)
```

Methods

getSelect

```
public SelectNode getSelect()
```

Returns the MDX parse tree behind this Query. The returned object is generated for each call to this function. Altering the returned SelectNode object won't affect the query itself.

Returns:
A SelectNode object representing the current query structure.

getCube

```
public Cube getCube()
```

Returns the underlying cube object that is used to query against.

Returns:
The Olap4j's Cube object.

getDimension

```
public QueryDimension getDimension(java.lang.String name)
```

Returns the Olap4j's Dimension object according to the name given as a parameter. If no dimension of the given name is found, a null value will be returned.

Parameters:
`name` - The name of the dimension you want the object for.

Returns:
The dimension object, null if no dimension of that name can be found.

swapAxes

```
public void swapAxes()
```

Swaps rows and columns axes. Only applicable if there are two axes.

getAxes

```
public java.util.Map getAxes()
```

Returns a map of the current query's axis.

Returns:
A standard Map object that represents the current query's axis.

getUnusedAxis

```
public QueryAxis getUnusedAxis()
```

Returns the fictional axis into which all unused dimensions are stored. All dimensions included in this axis will not be part of the query.

Returns:
The QueryAxis representing dimensions that are currently not used inside the query.

validate

```
public boolean validate()
  throws OlapException
```

execute

```
public CellSet execute()
  throws OlapException
```

Executes the query against the current OlapConnection and returns a CellSet object representation of the data.

Returns:
A proper CellSet object that represents the query execution results.

Throws:
`OlapException` - If something goes sour, an OlapException will be thrown to the caller. It could be caused by many things, like a stale connection. Look at the root cause for more details.

getName

```
public java.lang.String getName()
```

Returns this query's name. There is no guarantee that it is unique and is set at object instanciation.

Returns:
This query's name.

getLocale

```
public java.util.Locale getLocale()
```

Returns the current locale with which this query is expressed.

Returns:
A standard Locale object.

org.olap4j.query
Class QueryAxis

```
java.lang.Object
  |
  +-org.olap4j.query.QueryAxis
```

public class **QueryAxis**
extends java.lang.Object

An axis within an OLAP `Query`.

An axis has a location (columns, rows, etc) and has zero or more dimensions that are placed on it.

Constructor Summary

public	`QueryAxis(Query query, Axis location)` Creates a QueryAxis.

Method Summary

`java.util.List`	`getDimensions()` Returns a list of the dimensions placed on this QueryAxis.
`Axis`	`getLocation()` Returns the location of this `QueryAxis` in the query; `null` if unused.
`java.lang.String`	`getName()` Returns the name of this QueryAxis.
`boolean`	`isNonEmpty()` Returns whether this QueryAxis filters out empty rows.
`void`	`setNonEmpty(boolean nonEmpty)` Sets whether this QueryAxis filters out empty rows.

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructors

QueryAxis

```
public QueryAxis(Query query,
                 Axis location)
```

Creates a QueryAxis.

Parameters:
`query` - Query that the axis belongs to
`location` - Location of axis (e.g. ROWS, COLUMNS)

(continued from last page)

Methods

getLocation

`public Axis getLocation()`

Returns the location of this `QueryAxis` in the query; `null` if unused.

Returns:
location of this axis in the query

getDimensions

`public java.util.List getDimensions()`

Returns a list of the dimensions placed on this QueryAxis.

The list is mutable; you may call `getDimensions().clear()`, or `getDimensions().add(dimension)`, for instance. When a dimension is added to an axis, it is automatically removed from its previous axis.

Returns:
list of dimensions

getName

`public java.lang.String getName()`

Returns the name of this QueryAxis.

Returns:
the name of this axis, for example "ROWS", "COLUMNS".

isNonEmpty

`public boolean isNonEmpty()`

Returns whether this QueryAxis filters out empty rows. If true, axis filters out empty rows, and the MDX to evaluate the axis will be generated with the "NON EMPTY" expression.

Returns:
Whether this axis should filter out empty rows

See Also:
`setNonEmpty(boolean)`

setNonEmpty

`public void setNonEmpty(boolean nonEmpty)`

Sets whether this QueryAxis filters out empty rows.

Parameters:
`nonEmpty` - Whether this axis should filter out empty rows

See Also:
`isNonEmpty()`

org.olap4j.query
Class QueryDimension

```
java.lang.Object
  |
  +-org.olap4j.query.QueryDimension
```

public class **QueryDimension**
extends java.lang.Object

Usage of a dimension for an OLAP query.

It references an `Dimension` and allows the query creator to manage the member selections for the dimension. The state of a QueryDimension does not affect the Dimension object in any way so a single Dimension object can be referenced by many QueryDimension objects.

Nested Class Summary

class	`QueryDimension.SortOrder` QueryDimension.SortOrder

Constructor Summary

public	`QueryDimension(Query query, Dimension dimension)`

Method Summary

`Selection`	`createSelection(Member member)`
`Selection`	`createSelection(Member member, Selection.Operator operator)`
`QueryAxis`	`getAxis()`
`Dimension`	`getDimension()`
`java.lang.String`	`getName()`
`static java.lang.String[]`	`getNameParts(java.lang.String sel)`
`Query`	`getQuery()`
`java.util.List`	`getSelections()` Returns a list of the selections within this `QueryDimension`.
`QueryDimension.SortOrder`	`getSortOrder()`
`java.util.List`	`resolve(Selection selection)`
`void`	`setAxis(QueryAxis axis)`

void	`setDimension(Dimension dimension)`
void	`setSortOrder(QueryDimension.SortOrder order)`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructors

QueryDimension

```
public QueryDimension(Query query,
                      Dimension dimension)
```

Methods

getQuery

```
public Query getQuery()
```

setAxis

```
public void setAxis(QueryAxis axis)
```

getAxis

```
public QueryAxis getAxis()
```

getName

```
public java.lang.String getName()
```

createSelection

```
public Selection createSelection(Member member)
```

createSelection

```
public Selection createSelection(Member member,
          Selection.Operator operator)
```

getNameParts

```
public static java.lang.String[] getNameParts(java.lang.String sel)
```

resolve

```
public java.util.List resolve(Selection selection)
  throws OlapException
```

getSelections

```
public java.util.List getSelections()
```

Returns a list of the selections within this `QueryDimension`.

The list is mutable; you may call `getSelections().clear()`, or `getSelections().add(dimension)`, for instance.

Returns:
list of selections

getDimension

```
public Dimension getDimension()
```

setDimension

```
public void setDimension(Dimension dimension)
```

setSortOrder

```
public void setSortOrder(QueryDimension.SortOrder order)
```

getSortOrder

```
public QueryDimension.SortOrder getSortOrder()
```

org.olap4j.query
Class QueryDimension.SortOrder

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.query.QueryDimension.SortOrder
```

All Implemented Interfaces:

java.io.Serializable, java.lang.Comparable

public static final class **QueryDimension.SortOrder**
extends java.lang.Enum

Field Summary

public static final	ASC Ascending sort order.
public static final	DESC Descending sort order.

Method Summary

static QueryDimension.SortOrder	valueOf(java.lang.String name)
static QueryDimension.SortOrder[]	values()

Methods inherited from class java.lang.Enum

compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface java.lang.Comparable

compareTo

Fields

ASC

```
public static final org.olap4j.query.QueryDimension.SortOrder ASC
```

Ascending sort order.

DESC

`public static final org.olap4j.query.QueryDimension.SortOrder` **`DESC`**

Descending sort order.

Methods

values

`public final static QueryDimension.SortOrder[]` **`values`**`()`

valueOf

`public static QueryDimension.SortOrder` **`valueOf`**`(java.lang.String name)`

org.olap4j.query
Class RectangularCellSetFormatter

```
java.lang.Object
   |
   +-org.olap4j.query.RectangularCellSetFormatter
```

All Implemented Interfaces:
CellSetFormatter

---

public class **RectangularCellSetFormatter**
extends java.lang.Object
implements CellSetFormatter

Formatter that can convert a `CellSet` into a two-dimensional text layout.

With non-compact layout:

```
                         | 1997                                                |
                         | Q1                        | Q2                        |
                         |                           | 4                         |
                         | Unit Sales | Store Sales | Unit Sales | Store Sales |
----+----+---------------+------------+-------------+------------+-------------+
USA | CA | Los Angeles   |            |             |            |             |
    | WA | Seattle       |            |             |            |             |
    | CA | San Francisco |            |             |            |             |
```

With compact layout:

```
                     1997
                     Q1                     Q2
                                            4
                     Unit Sales Store Sales Unit Sales Store Sales
=== == ============= ========== =========== ========== ===========
USA CA Los Angeles           12        34.5         13       35.60
    WA Seattle               12        34.5         13       35.60
    CA San Francisco         12        34.5         13       35.60
```

This class is experimental. It is not part of the olap4j specification and is subject to change without notice.

Constructor Summary

public	`RectangularCellSetFormatter(boolean compact)` Creates a RectangularCellSetFormatter.

Method Summary

void	`format(CellSet cellSet, java.io.PrintWriter pw)`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.query.CellSetFormatter`
`format`

Constructors

RectangularCellSetFormatter

```
public RectangularCellSetFormatter(boolean compact)
```

Creates a RectangularCellSetFormatter.

Parameters:

`compact` - Whether to generate compact output

Methods

format

```
public void format(CellSet cellSet,
         java.io.PrintWriter pw)
```

org.olap4j.query
Interface Selection

public interface **Selection**
extends

A selection of members from an OLAP dimension hierarchy.

Concrete subclasses of this represent a real selection. Selections include things such as 'children of', 'siblings of', 'descendents of' etc.

This class is different from a `Member` because it represents an abstract member selection (e.g. children of widget' that may not represent any members whereas a Member represents a single member that is known to exist.

Nested Class Summary

`class`	`Selection.Operator` Selection.Operator

Method Summary

`Dimension`	`getDimension()`
`java.lang.String`	`getHierarchyName()`
`java.lang.String`	`getLevelName()`
`Member`	`getMember()`
`java.lang.String`	`getName()`
`Selection.Operator`	`getOperator()`
`void`	`setName(java.lang.String name)`
`void`	`setOperator(Selection.Operator operator)`

Methods

getName

`public java.lang.String` **`getName`**`()`

setName

`public void` **`setName`**`(java.lang.String name)`

(continued from last page)

getMember

```
public Member getMember()
```

getDimension

```
public Dimension getDimension()
```

getHierarchyName

```
public java.lang.String getHierarchyName()
```

getLevelName

```
public java.lang.String getLevelName()
```

getOperator

```
public Selection.Operator getOperator()
```

setOperator

```
public void setOperator(Selection.Operator operator)
```

org.olap4j.query
Class Selection.Operator

```
java.lang.Object
  |
  +-java.lang.Enum
      |
      +-org.olap4j.query.Selection.Operator
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

public static final class **Selection.Operator**
extends java.lang.Enum

Field Summary

public static final	ANCESTORS
public static final	CHILDREN
public static final	DESCENDANTS
public static final	INCLUDE_CHILDREN
public static final	MEMBER
public static final	SIBLINGS

Method Summary

static Selection.Operator	valueOf(java.lang.String name)
static Selection.Operator[]	values()

Methods inherited from class java.lang.Enum
compareTo, equals, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface java.lang.Comparable
compareTo

Fields

(continued from last page)

MEMBER

```
public static final org.olap4j.query.Selection.Operator MEMBER
```

CHILDREN

```
public static final org.olap4j.query.Selection.Operator CHILDREN
```

INCLUDE_CHILDREN

```
public static final org.olap4j.query.Selection.Operator INCLUDE_CHILDREN
```

SIBLINGS

```
public static final org.olap4j.query.Selection.Operator SIBLINGS
```

ANCESTORS

```
public static final org.olap4j.query.Selection.Operator ANCESTORS
```

DESCENDANTS

```
public static final org.olap4j.query.Selection.Operator DESCENDANTS
```

Methods

values

```
public final static Selection.Operator[] values()
```

valueOf

```
public static Selection.Operator valueOf(java.lang.String name)
```

org.olap4j.query
Class SelectionFactory

```
java.lang.Object
   |
   +-org.olap4j.query.SelectionFactory
```

public class **SelectionFactory**
extends java.lang.Object

Contains factory methods for creating implementations of `Selection`.

Created using `getSelectionFactory()`.

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

org.olap4j.query
Class TraditionalCellSetFormatter

```
java.lang.Object
  |
  +-org.olap4j.query.TraditionalCellSetFormatter
```

All Implemented Interfaces:

CellSetFormatter

public class **TraditionalCellSetFormatter**
extends java.lang.Object
implements CellSetFormatter

Formatter that can convert a `CellSet` into Mondrian's traditional layout.

This class is experimental. It is not part of the olap4j specification and is subject to change without notice.

Constructor Summary

public	`TraditionalCellSetFormatter()`

Method Summary

void	`format(CellSet cellSet, java.io.PrintWriter pw)`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.query.CellSetFormatter`

`format`

Constructors

TraditionalCellSetFormatter

```
public TraditionalCellSetFormatter()
```

Methods

format

```
public void format(CellSet cellSet,
                   java.io.PrintWriter pw)
```

Package
org.olap4j.transform

Provides services to transform MDX parse trees (experimental).

NOTE: This package is experimental. Classes may be renamed or removed in a future release of olap4j.

org.olap4j.transform
Class AxisTransform

```
java.lang.Object
  |
  +-org.olap4j.transform.AxisTransform
```

All Implemented Interfaces:

MdxQueryTransform

Direct Known Subclasses:

DrillDownOnPositionTransform, DrillReplaceTransform, RollUpLevelTransform

---

public abstract class **AxisTransform**
extends java.lang.Object
implements MdxQueryTransform

Abstract representation of an MDX query transform acting on a single query axis (e.g. drill-down on member, roll-up, ...)

Method Summary

`SelectNode`	`apply(SelectNode sn)`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.transform.MdxQueryTransform`
`apply, getDescription, getName`

Methods

apply

`public SelectNode apply(SelectNode sn)`

org.olap4j.transform
Class DrillDownOnPositionTransform

```
java.lang.Object
  |
  +-org.olap4j.transform.AxisTransform
      |
      +-org.olap4j.transform.DrillDownOnPositionTransform
```

All Implemented Interfaces:
MdxQueryTransform

---

public class **DrillDownOnPositionTransform**
extends AxisTransform

Drill down on position transform TODO: transform to be completed, not working for now.

Description: Adds the children of a member at a specific position on an axis. The member to drill is identified from a CellSet with the axis, positionOrdinalInAxis and memberOrdinalInPosition arguments. The drilled member will still be present on the axis, in addition to its children. It is recommended to apply a Hierarchize transform to the same axis of the resulting query, in order to have members in correct hierarchical order.

Example of use: the user clicks on a member in a crosstab axis, in order to see its children in addition to the member itself.

Applicability: this transform is applicable only to members in a query that are drillable, i.e. non-leaf members. The CellSet resulting from the execution of the initial MDX query must also be available.

Constructor Summary

public	`DrillDownOnPositionTransform(Axis axis, int positionOrdinalInAxis, int memberOrdinalInPosition, CellSet cellSet)` ctor

Method Summary

`java.lang.String`	`getDescription()`
`java.lang.String`	`getName()`

Methods inherited from class `org.olap4j.transform.AxisTransform`

`apply`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.transform.MdxQueryTransform`

`apply, getDescription, getName`

Constructors

(continued from last page)

DrillDownOnPositionTransform

```
public DrillDownOnPositionTransform(Axis axis,
                                    int positionOrdinalInAxis,
                                    int memberOrdinalInPosition,
                                    CellSet cellSet)
```

ctor

Parameters:

`axis`
`positionOrdinalInAxis`
`memberOrdinalInPosition`
`cellSet`

Methods

getName

```
public java.lang.String getName()
```

getDescription

```
public java.lang.String getDescription()
```

org.olap4j.transform
Class DrillReplaceTransform

```
java.lang.Object
  |
  +-org.olap4j.transform.AxisTransform
      |
      +-org.olap4j.transform.DrillReplaceTransform
```

All Implemented Interfaces:
MdxQueryTransform

public class **DrillReplaceTransform**
extends AxisTransform

Drill replace transformation

Description: Replaces a member at a specific position on an axis by its children. The member to drill is identified from a CellSet with the axis, positionOrdinalInAxis and memberOrdinalInPosition arguments.

Example of use: the user clicks on a member in a crosstab axis, in order to see its children.

Applicability: this transform is applicable only to members in a query that are drillable, i.e. non-leaf members. The CellSet resulting from the execution of the initial MDX query must also be available.

Constructor Summary

`public`	`DrillReplaceTransform(Axis axis, int positionOrdinalInAxis, int memberOrdinalInPosition, CellSet cellSet)` ctor

Method Summary

`java.lang.String`	`getDescription()`
`java.lang.String`	`getName()`

Methods inherited from class `org.olap4j.transform.AxisTransform`

`apply`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.transform.MdxQueryTransform`

`apply, getDescription, getName`

Constructors

(continued from last page)

DrillReplaceTransform

```
public DrillReplaceTransform(Axis axis,
                             int positionOrdinalInAxis,
                             int memberOrdinalInPosition,
                             CellSet cellSet)
```

ctor

Parameters:

`axis` - axis (of the resulting CellSet) the member to be drilled
`positionOrdinalInAxis` - position ordinal in axis of the member to be drilled
`memberOrdinalInPosition` - ordinal in position of the member to be drilled
`cellSet` - the CellSet resulting from execution of the query to be transformed

Methods

getName

```
public java.lang.String getName()
```

getDescription

```
public java.lang.String getDescription()
```

org.olap4j.transform
Interface MdxQueryTransform

All Known Implementing Classes:
AxisTransform

public interface **MdxQueryTransform**
extends

MDX Query Transformation

General interface for transforming an MDX query to another one, according to behavior and parameters encapsulated in implementing classes

Method Summary

`SelectNode`	`apply(SelectNode sn)`
`java.lang.String`	`getDescription()`
`java.lang.String`	`getName()`

Methods

getName

```
public java.lang.String getName()
```

getDescription

```
public java.lang.String getDescription()
```

apply

```
public SelectNode apply(SelectNode sn)
```

org.olap4j.transform
Class Quax

```
java.lang.Object
  |
  +-org.olap4j.transform.Quax
```

public class **Quax**
extends java.lang.Object

Representation of member expressions on a query axis, derived from CellSetAxis objects.

Quaxes are used by MDX axis query transforms, to construct and use an internal tree-like representation of positions and members from the result CellSetAxis objects of a previous MDX query. This is needed for OLAP navigation operators like drill-down on a position.

Inspired from the JPivot Quax class.

NOTE: not exactly sure how to implement this, to be completed...

Constructor Summary

`public`	`Quax(CellSetAxis cellSetAxis)`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructors

Quax

`public Quax(CellSetAxis cellSetAxis)`

org.olap4j.transform
Class RollUpLevelTransform

```
java.lang.Object
  |
  +-org.olap4j.transform.AxisTransform
      |
      +-org.olap4j.transform.RollUpLevelTransform
```

All Implemented Interfaces:
MdxQueryTransform

public class **RollUpLevelTransform**
extends AxisTransform

Roll-up level transformation

Description: Replaces a member at a specific position on an axis by all the members of its parent's level. The member to roll-up is identified from a CellSet with the axis, positionOrdinalInAxis and memberOrdinalInPosition arguments.

Example of use: the user clicks on a member in a crosstab axis, in order to roll up to the members of the upper level.

Applicability: this transform is applicable only to members in a query that are have a parent. (Note: how would this work in parent-child hierarchies?)

Constructor Summary

public	`RollUpLevelTransform(Axis axis, int positionOrdinalInAxis, int memberOrdinalInPosition, CellSet cellSet)` ctor

Method Summary

`java.lang.String`	`getDescription()`
`java.lang.String`	`getName()`

Methods inherited from class `org.olap4j.transform.AxisTransform`

`apply`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.transform.MdxQueryTransform`

`apply, getDescription, getName`

Constructors

(continued from last page)

RollUpLevelTransform

```
public RollUpLevelTransform(Axis axis,
                            int positionOrdinalInAxis,
                            int memberOrdinalInPosition,
                            CellSet cellSet)
```

ctor

Parameters:
`axis`
`positionOrdinalInAxis`
`memberOrdinalInPosition`
`cellSet`

Methods

getName

```
public java.lang.String getName()
```

getDescription

```
public java.lang.String getDescription()
```

org.olap4j.transform
Class StandardTransformLibrary

```
java.lang.Object
  |
  +-org.olap4j.transform.StandardTransformLibrary
```

public class **StandardTransformLibrary**
extends java.lang.Object

Standard transformations library NOTE: is this really needed since transforms' ctors have the same parameters as these functions? This serves only as a place to conveniently regroup transforms in a "library".

Constructor Summary

public	StandardTransformLibrary()

Method Summary

static MdxQueryTransform	createDrillDownOnPositionTransform(Axis axis, int positionOrdinalInAxis, int memberOrdinalInPosition, CellSet cellSet)
static MdxQueryTransform	createDrillReplaceTransform(Axis axis, int positionOrdinalInAxis, int memberOrdinalInPosition, CellSet cellSet)
static MdxQueryTransform	createRollUpLevelTransform(Axis axis, int positionOrdinalInAxis, int memberOrdinalInPosition, CellSet cellSet)

Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

StandardTransformLibrary

```
public StandardTransformLibrary()
```

Methods

createDrillReplaceTransform

```
public static MdxQueryTransform createDrillReplaceTransform(Axis axis,
         int positionOrdinalInAxis,
         int memberOrdinalInPosition,
         CellSet cellSet)
```

(continued from last page)

createDrillDownOnPositionTransform

```
public static MdxQueryTransform createDrillDownOnPositionTransform(Axis axis,
         int positionOrdinalInAxis,
         int memberOrdinalInPosition,
         CellSet cellSet)
```

createRollUpLevelTransform

```
public static MdxQueryTransform createRollUpLevelTransform(Axis axis,
         int positionOrdinalInAxis,
         int memberOrdinalInPosition,
         CellSet cellSet)
```

Package
org.olap4j.type

Type system for MDX expressions.

org.olap4j.type
Class BooleanType

```
java.lang.Object
  |
  +-org.olap4j.type.ScalarType
      |
      +-org.olap4j.type.BooleanType
```

All Implemented Interfaces:
Type

public class **BooleanType**
extends ScalarType

The type of a boolean (logical) expression.

An example of a boolean expression is the predicate`[Measures].[Unit Sales] > 1000`

Constructor Summary

`public`	`BooleanType()` Creates a boolean type.

Method Summary

`java.lang.String`	`toString()`

Methods inherited from class `org.olap4j.type.ScalarType`

`getDimension, getHierarchy, getLevel, usesDimension`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.type.Type`

`getDimension, getHierarchy, getLevel, usesDimension`

Constructors

BooleanType

`public` **`BooleanType`**`()`

Creates a boolean type.

Methods

(continued from last page)

toString

```
public java.lang.String toString()
```

org.olap4j.type
Class CubeType

```
java.lang.Object
  |
  +-org.olap4j.type.CubeType
```

All Implemented Interfaces:
Type

public class **CubeType**
extends java.lang.Object
implements Type

The type of an expression which represents a Cube or Virtual Cube.

Constructor Summary

`public`	`CubeType(Cube cube)` Creates a type representing a cube.

Method Summary

`boolean`	`equals(java.lang.Object obj)`
`Cube`	`getCube()` Returns the cube.
`Dimension`	`getDimension()`
`Hierarchy`	`getHierarchy()`
`Level`	`getLevel()`
`int`	`hashCode()`
`boolean`	`usesDimension(Dimension dimension, boolean maybe)`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.type.Type`

`getDimension, getHierarchy, getLevel, usesDimension`

Constructors

(continued from last page)

CubeType

```
public CubeType(Cube cube)
```

Creates a type representing a cube.

Parameters:
`cube` - Cube

Methods

getCube

```
public Cube getCube()
```

Returns the cube.

Returns:
the cube

usesDimension

```
public boolean usesDimension(Dimension dimension,
         boolean maybe)
```

getDimension

```
public Dimension getDimension()
```

getHierarchy

```
public Hierarchy getHierarchy()
```

getLevel

```
public Level getLevel()
```

equals

```
public boolean equals(java.lang.Object obj)
```

hashCode

```
public int hashCode()
```

org.olap4j.type
Class DecimalType

```
java.lang.Object
  |
  +-org.olap4j.type.ScalarType
      |
      +-org.olap4j.type.NumericType
          |
          +-org.olap4j.type.DecimalType
```

All Implemented Interfaces:
Type

public class **DecimalType**
extends NumericType

Subclass of `NumericType` which guarantees fixed number of decimal places. In particular, a decimal with zero scale is an integer.

Constructor Summary

public	`DecimalType(int precision, int scale)` Creates a decimal type with precision and scale.

Method Summary

int	`getPrecision()` Returns the maximum number of decimal digits which a value of this type can have.
int	`getScale()` Returns the number of digits to the right of the decimal point.
java.lang.String	`toString()`

Methods inherited from class `org.olap4j.type.NumericType`

`toString`

Methods inherited from class `org.olap4j.type.ScalarType`

`getDimension, getHierarchy, getLevel, usesDimension`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.type.Type`

`getDimension, getHierarchy, getLevel, usesDimension`

Constructors

(continued from last page)

DecimalType

```
public DecimalType(int precision,
                   int scale)
```

Creates a decimal type with precision and scale.

Examples:

- 123.45 has precision 5, scale 2.
- 12,345,000 has precision 5, scale -3.

The largest value is 10 ^ (precision - scale). Hence the largest `DECIMAL(5, -3)` value is 10 ^ 8.

Parameters:

`precision` - Maximum number of decimal digits which a value of this type can have. Must be greater than zero. Use `Integer.MAX_VALUE` if the precision is unbounded.
`scale` - Number of digits to the right of the decimal point.

Methods

getPrecision

```
public int getPrecision()
```

Returns the maximum number of decimal digits which a value of this type can have.

Returns:

maximum precision allowed for values of this type

getScale

```
public int getScale()
```

Returns the number of digits to the right of the decimal point.

Returns:

number of digits to the right of the decimal point

toString

```
public java.lang.String toString()
```

org.olap4j.type
Class DimensionType

```
java.lang.Object
  |
  +-org.olap4j.type.DimensionType
```

All Implemented Interfaces:

Type

public class **DimensionType**
extends java.lang.Object
implements Type

The type of an expression which represents a Dimension.

Field Summary

public static final	Unknown

Constructor Summary

public	DimensionType(Dimension dimension) Creates a type representing a dimension.

Method Summary

Dimension	getDimension()
Hierarchy	getHierarchy()
Level	getLevel()
java.lang.String	toString()
boolean	usesDimension(Dimension dimension, boolean maybe)

Methods inherited from class `java.lang.Object`

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface `org.olap4j.type.Type`

getDimension, getHierarchy, getLevel, usesDimension

Fields

(continued from last page)

Unknown

```
public static final org.olap4j.type.DimensionType Unknown
```

Constructors

DimensionType

```
public DimensionType(Dimension dimension)
```

Creates a type representing a dimension.

Parameters:

`dimension` - Dimension which values of this type must belong to, or null if not known

Methods

usesDimension

```
public boolean usesDimension(Dimension dimension,
         boolean maybe)
```

getHierarchy

```
public Hierarchy getHierarchy()
```

getLevel

```
public Level getLevel()
```

getDimension

```
public Dimension getDimension()
```

toString

```
public java.lang.String toString()
```

org.olap4j.type
Class HierarchyType

```
java.lang.Object
   |
   +-org.olap4j.type.HierarchyType
```

All Implemented Interfaces:

Type

public class **HierarchyType**
extends java.lang.Object
implements Type

The type of an expression which represents a hierarchy.

Constructor Summary

public	`HierarchyType(Dimension dimension, Hierarchy hierarchy)` Creates a type representing a hierarchy.

Method Summary

`Dimension`	`getDimension()`
`Hierarchy`	`getHierarchy()`
`Level`	`getLevel()`
`java.lang.String`	`toString()`
`boolean`	`usesDimension(Dimension dimension, boolean maybe)`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.type.Type`

`getDimension, getHierarchy, getLevel, usesDimension`

Constructors

HierarchyType

```
public HierarchyType(Dimension dimension,
                     Hierarchy hierarchy)
```

Creates a type representing a hierarchy.

(continued from last page)

Parameters:

`dimension` - Dimension which values of this type must belong to, or null if not known
`hierarchy` - Hierarchy which values of this type must belong to, or null if not known

Methods

usesDimension

```
public boolean usesDimension(Dimension dimension,
         boolean maybe)
```

getDimension

```
public Dimension getDimension()
```

getHierarchy

```
public Hierarchy getHierarchy()
```

getLevel

```
public Level getLevel()
```

toString

```
public java.lang.String toString()
```

org.olap4j.type
Class LevelType

```
java.lang.Object
  |
  +-org.olap4j.type.LevelType
```

All Implemented Interfaces:

Type

public class **LevelType**
extends java.lang.Object
implements Type

The type of an expression which represents a level.

Constructor Summary

public	`LevelType(Dimension dimension, Hierarchy hierarchy, Level level)` Creates a type representing a level.

Method Summary

`Dimension`	`getDimension()`
`Hierarchy`	`getHierarchy()`
`Level`	`getLevel()`
`java.lang.String`	`toString()`
`boolean`	`usesDimension(Dimension dimension, boolean maybe)`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.type.Type`

`getDimension, getHierarchy, getLevel, usesDimension`

Constructors

LevelType

```
public LevelType(Dimension dimension,
                 Hierarchy hierarchy,
                 Level level)
```

Creates a type representing a level.

(continued from last page)

Parameters:

`dimension` - Dimension which values of this type must belong to, or null if not known
`hierarchy` - Hierarchy which values of this type must belong to, or null if not known
`level` - Level which values of this type must belong to, or null if not known

Methods

usesDimension

```
public boolean usesDimension(Dimension dimension,
         boolean maybe)
```

getDimension

```
public Dimension getDimension()
```

getHierarchy

```
public Hierarchy getHierarchy()
```

getLevel

```
public Level getLevel()
```

toString

```
public java.lang.String toString()
```

org.olap4j.type
Class MemberType

```
java.lang.Object
  └-org.olap4j.type.MemberType
```

All Implemented Interfaces:
Type

public class **MemberType**
extends java.lang.Object
implements Type

The type of an expression which represents a member.

Constructor Summary

public	`MemberType(Dimension dimension, Hierarchy hierarchy, Level level, Member member)` Creates a type representing a member.

Method Summary

`Dimension`	`getDimension()`
`Hierarchy`	`getHierarchy()`
`Level`	`getLevel()`
`Member`	`getMember()` Returns the member of this type, or null if not known.
`java.lang.String`	`toString()`
`boolean`	`usesDimension(Dimension dimension, boolean maybe)`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.type.Type`

`getDimension, getHierarchy, getLevel, usesDimension`

Constructors

(continued from last page)

MemberType

```
public MemberType(Dimension dimension,
                  Hierarchy hierarchy,
                  Level level,
                  Member member)
```

Creates a type representing a member.

Parameters:

`dimension` - Dimension the member belongs to, or null if not known.
`hierarchy` - Hierarchy the member belongs to, or null if not known.
`level` - Level the member belongs to, or null if not known
`member` - The precise member, or null if not known

Methods

toString

```
public java.lang.String toString()
```

getHierarchy

```
public Hierarchy getHierarchy()
```

getLevel

```
public Level getLevel()
```

getMember

```
public Member getMember()
```

Returns the member of this type, or null if not known.

Returns:

member of this type

usesDimension

```
public boolean usesDimension(Dimension dimension,
         boolean maybe)
```

getDimension

```
public Dimension getDimension()
```

org.olap4j.type
Class NullType

```
java.lang.Object
  |
  +-org.olap4j.type.ScalarType
      |
      +-org.olap4j.type.NullType
```

All Implemented Interfaces:

Type

---

public class **NullType**
extends ScalarType

The type of a null expression.

Constructor Summary

public	`NullType()` Creates a null type.

Method Summary

`java.lang.String`	`toString()`

Methods inherited from class `org.olap4j.type.ScalarType`
`getDimension, getHierarchy, getLevel, usesDimension`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.type.Type`
`getDimension, getHierarchy, getLevel, usesDimension`

Constructors

NullType

`public NullType()`

Creates a null type.

Methods

toString

`public java.lang.String toString()`

(continued from last page)

org.olap4j.type
Class NumericType

```
java.lang.Object
  |
  +-org.olap4j.type.ScalarType
      |
      +-org.olap4j.type.NumericType
```

All Implemented Interfaces:
Type

Direct Known Subclasses:
DecimalType

public class **NumericType**
extends ScalarType

The type of a numeric expression.

Constructor Summary

`public`	`NumericType()` Creates a numeric type.

Method Summary

`java.lang.String`	`toString()`

Methods inherited from class `org.olap4j.type.ScalarType`

`getDimension, getHierarchy, getLevel, usesDimension`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.type.Type`

`getDimension, getHierarchy, getLevel, usesDimension`

Constructors

NumericType

`public NumericType()`

Creates a numeric type.

Methods

(continued from last page)

toString

```
public java.lang.String toString()
```

org.olap4j.type
Class ScalarType

```
java.lang.Object
  |
  +-org.olap4j.type.ScalarType
```

All Implemented Interfaces:
Type

Direct Known Subclasses:
BooleanType, NullType, NumericType, StringType, SymbolType

public class **ScalarType**
extends java.lang.Object
implements Type

Base class for types which represent scalar values.

An instance of this class means a scalar value of unknown type. Usually one of the derived classes `NumericType`, `StringType`, `BooleanType` is used instead.

Constructor Summary

public	ScalarType()

Method Summary

Dimension	getDimension()
Hierarchy	getHierarchy()
Level	getLevel()
boolean	usesDimension(Dimension dimension, boolean maybe)

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.type.Type`

`getDimension, getHierarchy, getLevel, usesDimension`

Constructors

ScalarType

`public ScalarType()`

(continued from last page)

Methods

usesDimension

```
public boolean usesDimension(Dimension dimension,
         boolean maybe)
```

getHierarchy

```
public Hierarchy getHierarchy()
```

getLevel

```
public Level getLevel()
```

getDimension

```
public Dimension getDimension()
```

org.olap4j.type
Class SetType

```
java.lang.Object
  |
  +-org.olap4j.type.SetType
```

All Implemented Interfaces:

Type

public class **SetType**
extends java.lang.Object
implements Type

Set type.

Constructor Summary

public	`SetType(Type elementType)` Creates a type representing a set of elements of a given type.

Method Summary

`Dimension`	`getDimension()`
`Type`	`getElementType()` Returns the type of the elements of this set.
`Hierarchy`	`getHierarchy()`
`Level`	`getLevel()`
`boolean`	`usesDimension(Dimension dimension, boolean maybe)`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.type.Type`

`getDimension, getHierarchy, getLevel, usesDimension`

Constructors

SetType

`public SetType(Type elementType)`

Creates a type representing a set of elements of a given type.

Parameters:

(continued from last page)

`elementType` - The type of the elements in the set, or null if not known

Methods

getElementType

```
public Type getElementType()
```

Returns the type of the elements of this set.

Returns:
element type

usesDimension

```
public boolean usesDimension(Dimension dimension,
          boolean maybe)
```

getDimension

```
public Dimension getDimension()
```

getHierarchy

```
public Hierarchy getHierarchy()
```

getLevel

```
public Level getLevel()
```

org.olap4j.type
Class StringType

```
java.lang.Object
  |
  +-org.olap4j.type.ScalarType
      |
      +-org.olap4j.type.StringType
```

All Implemented Interfaces:
Type

public class **StringType**
extends ScalarType

The type of a string expression.

Constructor Summary

public	`StringType()` Creates a string type.

Methods inherited from class `org.olap4j.type.ScalarType`
`getDimension, getHierarchy, getLevel, usesDimension`

Methods inherited from class `java.lang.Object`
`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.type.Type`
`getDimension, getHierarchy, getLevel, usesDimension`

Constructors

StringType

`public StringType()`

Creates a string type.

org.olap4j.type
Class SymbolType

```
java.lang.Object
  |
  +-org.olap4j.type.ScalarType
      |
      +-org.olap4j.type.SymbolType
```

All Implemented Interfaces:
Type

public class **SymbolType**
extends ScalarType

The type of a symbolic expression.

Symbols are identifiers which occur in particular function calls, generally to indicate an option for how the function should be executed. They are similar to an enumerated type in other languages.

For example, the optional 3rd argument to the `Order` function can be one of the symbols `ASC`, `DESC`, `BASC`, `BDESC`. The signature of the `Order` function is therefore`Order(<Set>, <Scalar expression> [, <Symbol>])`and `Order([Store].Members, [Measures].[Unit Sales], BDESC)`would be a valid call to the function.

Constructor Summary

`public`	`SymbolType()` Creates a symbol type.

Methods inherited from class `org.olap4j.type.ScalarType`

`getDimension, getHierarchy, getLevel, usesDimension`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.type.Type`

`getDimension, getHierarchy, getLevel, usesDimension`

Constructors

SymbolType

`public SymbolType()`

Creates a symbol type.

org.olap4j.type
Class TupleType

```
java.lang.Object
   |
   +-org.olap4j.type.TupleType
```

All Implemented Interfaces:

Type

---

public class **TupleType**
extends java.lang.Object
implements Type

Tuple type.

Constructor Summary

`public`	`TupleType(Type[] elementTypes)` Creates a type representing a tuple whose fields are the given types.

Method Summary

`Dimension`	`getDimension()`
`Hierarchy`	`getHierarchy()`
`Level`	`getLevel()`
`java.lang.String`	`toString()`
`boolean`	`usesDimension(Dimension dimension, boolean maybe)`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `org.olap4j.type.Type`

`getDimension, getHierarchy, getLevel, usesDimension`

Constructors

TupleType

```
public TupleType(Type[] elementTypes)
```

Creates a type representing a tuple whose fields are the given types.

Parameters:

(continued from last page)

`elementTypes` - Array of field types

Methods

toString

```
public java.lang.String toString()
```

usesDimension

```
public boolean usesDimension(Dimension dimension,
         boolean maybe)
```

getDimension

```
public Dimension getDimension()
```

getHierarchy

```
public Hierarchy getHierarchy()
```

getLevel

```
public Level getLevel()
```

org.olap4j.type
Interface Type

All Known Implementing Classes:
CubeType, DimensionType, HierarchyType, LevelType, MemberType, ScalarType, SetType, TupleType

public interface **Type**
extends

Type of an MDX expression.

All type objects are immutable.

Method Summary

`Dimension`	`getDimension()` Returns the dimension of this type, or null if not known.
`Hierarchy`	`getHierarchy()` Returns the hierarchy of this type.
`Level`	`getLevel()` Returns the level of this type, or null if not known.
`boolean`	`usesDimension(Dimension dimension, boolean maybe)` Returns whether this type contains a given dimension.

Methods

usesDimension

```
public boolean usesDimension(Dimension dimension,
          boolean maybe)
```

Returns whether this type contains a given dimension.

For example:

- `DimensionType([Gender])` uses only the `[Gender]` dimension.
- `TupleType(MemberType([Gender]), MemberType([Store]))` uses `[Gender]` and `[Store]` dimensions.

The `maybe` parameter comes into play when the dimensional information is incomplete. For example, when applied to `TupleType(MemberType(null), MemberType([Store]))`, `usesDimension([Gender], false)` returns true because it is possible that the expression returns a member of the `[Gender]` dimension.

Parameters:

`dimension` - Dimension

`maybe` - If true, returns true only if this type definitely uses the dimension

Returns:

whether this type definitely (or if `maybe` is true, possibly) uses the given dimension

getDimension

```
public Dimension getDimension()
```

Returns the dimension of this type, or null if not known.

Returns:
dimension of this type

getHierarchy

```
public Hierarchy getHierarchy()
```

Returns the hierarchy of this type. If not applicable, throws.

Returns:
hierarchy of this type

getLevel

```
public Level getLevel()
```

Returns the level of this type, or null if not known.

Returns:
level of this type

org.olap4j.type
Class TypeUtil

```
java.lang.Object
  |
  +-org.olap4j.type.TypeUtil
```

public class **TypeUtil**
extends java.lang.Object

Utility methods relating to types.

NOTE: This class is experimental. Not part of the public olap4j API.

Constructor Summary

public	TypeUtil()

Method Summary

static boolean	canEvaluate(Type type) Returns whether a value of a given type can be evaluated to a scalar value.
static boolean	isSet(Type type) Returns whether a type is a set type.

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructors

TypeUtil

`public TypeUtil()`

Methods

canEvaluate

`public static boolean canEvaluate(Type type)`

(continued from last page)

Returns whether a value of a given type can be evaluated to a scalar value.

The rules are as follows:

- Clearly boolean, numeric and string expressions can be evaluated.
- Member and tuple expressions can be interpreted as a scalar value. The expression is evaluated to establish the context where a measure can be evaluated.
- Hierarchy and dimension expressions are implicitly converted into the current member, and evaluated as above.
- Level expressions cannot be evaluated
- Cube and Set (even sets with a single member) cannot be evaluated.

Parameters:

`type` - Type

Returns:

Whether an expression of this type can be evaluated to yield a scalar value.

isSet

```
public static boolean isSet(Type type)
```

Returns whether a type is a set type.

Parameters:

`type` - Type

Returns:

Whether a value of this type can be evaluated to yield a set.