DataException

Method Summery

Number	#getErrorCode() Returns the error code for this ServoyException.
String	#getMessage() Returns the string message for this ServoyException.
Object[]	#getParameters() Returns the parameters of the SQL query that caused this DataException in an array.
String	#getSQL() Returns the SQL query that caused this DataException.
String	#getSQLState() Returns the SQLState for this DataException.
Object	#getValue() Returns the value for this DataException.
Number	#getVendorErrorCode() Returns the error code of the error thrown by the back-end database server.

Method Details

getErrorCode

Number **getErrorCode**()

Returns the error code for this ServoyException. Can be one of the constants declared in ServoyException.

Returns

Number – the error code for this ServoyException. Can be one of the constants declared in ServoyException.

Sample

```
//this sample script should be attached to onError method handler in the solution settings
var e = arguments[0];
application.output("Exception Object: "+e)
application.output("MSG: "+e.getMessage())
if (e instanceof ServoyException)
{
	application.output("is a ServoyException")
	application.output("Errorcode: "+e.getErrorCode())
	if (e.getErrorCode() == ServoyException.SAVE_FAILED)
	{
		plugins.dialogs.showErrorDialog( "Error",  "It seems you did not fill in a required field",
'OK');
		//Get the failed records after a save
		var array = databaseManager.getFailedRecords()
		for( var i = 0 ; i < array.length ; i++ )
		{
			var record = array[i];
			application.output(record.exception);
			if (record.exception instanceof DataException)
			{
				application.output("SQL: "+record.exception.getSQL())
				application.output("SQLState: "+record.exception.getSQLState())
				application.output("VendorErrorCode: "+record.exception.getVendorErrorCode())
			}
		}
		return false
	}
}
//if returns false or no return, error is not reported to client; if returns true error is reported
//by default error report means logging the error, in smart client an error dialog will also show up
return true
```

getMessage

String **getMessage**()

Returns the string message for this ServoyException.

Returns

String – the string message for this ServoyException.

Sample

```
//this sample script should be attached to onError method handler in the solution settings
var e = arguments[0];
application.output("Exception Object: "+e)
application.output("MSG: "+e.getMessage())
if (e instanceof ServoyException)
{
	application.output("is a ServoyException")
	application.output("Errorcode: "+e.getErrorCode())
	if (e.getErrorCode() == ServoyException.SAVE_FAILED)
	{
		plugins.dialogs.showErrorDialog( "Error",  "It seems you did not fill in a required field",
'OK');
		//Get the failed records after a save
		var array = databaseManager.getFailedRecords()
		for( var i = 0 ; i < array.length ; i++ )
		{
			var record = array[i];
			application.output(record.exception);
			if (record.exception instanceof DataException)
			{
				application.output("SQL: "+record.exception.getSQL())
				application.output("SQLState: "+record.exception.getSQLState())
				application.output("VendorErrorCode: "+record.exception.getVendorErrorCode())
			}
		}
		return false
	}
}
//if returns false or no return, error is not reported to client; if returns true error is reported
//by default error report means logging the error, in smart client an error dialog will also show up
return true
```

getParameters

Object[] **getParameters**()

Returns the parameters of the SQL query that caused this DataException in an array.

Returns

Object[] – the parameters of the SQL query that caused this DataException in an array.

Sample

```
var record = array[i];
application.output(record.exception);
if (record.exception instanceof DataException)
{
	var param = record.exception.getParameters();
	for (j = 0; j < param.length; j++)
	{
		application.output("SQL Parameter [" + j + "]: " + param[j]);
	}
}
```

getSQL

String **getSQL**()

Returns the SQL query that caused this DataException.

Returns

String – the SQL query that caused this DataException.

Sample

```
var record = array[i];
application.output(record.exception);
if (record.exception instanceof DataException)
{
        application.output("SQL: " + record.exception.getSQL());
}
```

getSQLState

String **getSQLState**()

Returns the SQLState for this DataException.
This is a "SQLstate" string, which follows either the XOPEN SQLstate conventions or the SQL 99 conventions.
The values of the SQLState string are described in the appropriate spec.

Returns

String – the SQLState for this DataException.

Sample

```
var record = array[i];
application.output(record.exception);
if (record.exception instanceof DataException)
{
        application.output("SQLState: " + record.exception.getSQLState());
}
```

getValue

Object **getValue**()

Returns the value for this DataException.
The value is the object thrown in table pre-insert, pre-update or pre-delete triggers.

Returns

Object – the value for this DataException.

Sample

```
var record = array[i];
application.output(record.exception);
if (record.exception instanceof DataException)
{
        application.output("VALUE: " + record.exception.getValue());
}
```

getVendorErrorCode

Number **getVendorErrorCode**()

Returns the error code of the error thrown by the back-end database server.

Returns

Number – the error code of the error thrown by the back-end database server.

Sample

```
var record = array[i];
application.output(record.exception);
if (record.exception instanceof DataException)
{
        application.output("VendorErrorCode: " + record.exception.getVendorErrorCode());
}
```