

ServoyException

Return Types

[DataException](#)

Constants Summary

Number	#ABSTRACT_FORM Exception code for ABSTRACT_FORM.
Number	#ACQUIRE_LOCK_FAILURE Exception code for ACQUIRE_LOCK_FAILURE.
Number	#BAD_SQL_SYNTAX Exception code for BAD_SQL_SYNTAX.
Number	#CLIENT_NOT_AUTHORIZED Exception code for CLIENT_NOT_AUTHORIZED.
Number	#DATA_ACCESS_RESOURCE_FAILURE Exception code for DATA_ACCESS_RESOURCE_FAILURE.
Number	#DATA_INTEGRITY_VIOLATION Exception code for DATA_INTEGRITY_VIOLATION.
Number	#DEADLOCK Exception code for DEADLOCK.
Number	#DELETE_NOT_GRANTED Exception code for DELETE_NOT_GRANTED.
Number	#EXECUTE_PROGRAM_FAILED Exception code for EXECUTE_PROGRAM_FAILED.
Number	#INCORRECT_LOGIN Exception code for INCORRECT_LOGIN.
Number	#INVALID_INPUT Exception code for INVALID_INPUT.
Number	#INVALID_RESULTSET_ACCESS Exception code for INVALID_RESULTSET_ACCESS.
Number	#MAINTENANCE_MODE Exception code for MAINTENANCE_MODE.
Number	#NO_ACCESS Exception code for NO_ACCESS.
Number	#NO_CREATE_ACCESS Exception code for NO_CREATE_ACCESS.
Number	#NO_DELETE_ACCESS Exception code for NO_DELETE_ACCESS.
Number	#NO_LICENSE Exception code for NO_LICENSE.
Number	#NO_MODIFY_ACCESS Exception code for NO_MODIFY_ACCESS.
Number	#NO_PARENT_DELETE_WITH_RELATED_RECORDS Exception code for NO_PARENT_DELETE_WITH_RELATED_RECORDS.
Number	#NO_RELATED_CREATE_ACCESS Exception code for NO_RELATED_CREATE_ACCESS.
Number	#PERMISSION_DENIED Exception code for PERMISSION_DENIED.
Number	#RECORD_LOCKED Exception code for RECORD_LOCKED.
Number	#RECORD_VALIDATION_FAILED Exception code for RECORD_VALIDATION_FAILED.
Number	#SAVE_FAILED Exception code for SAVE_FAILED.
Number	#UNEXPECTED_UPDATE_COUNT Exception code for UNEXPECTED_UPDATE_COUNT.
Number	#UNKNOWN_DATABASE_EXCEPTION Exception code for UNKNOWN_DATABASE_EXCEPTION.

Method Summary

Number [#getErrorCode\(\)](#)
Returns the error code for this ServoyException.

String [#getMessage\(\)](#)
Returns the string message for this ServoyException.

String [#getStackTrace\(\)](#)
Returns the stack trace for this ServoyException.

Constants Details

ABSTRACT_FORM

Exception code for ABSTRACT_FORM.

This code is used when a form, that cannot be created, is shown (for example, a form without parts).

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

ACQUIRE_LOCK_FAILURE

Exception code for ACQUIRE_LOCK_FAILURE.

This code is used when a database failed to lock a row or table.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

BAD_SQL_SYNTAX

Exception code for BAD_SQL_SYNTAX.

This code is used when a database exception is recognized as an sql syntax error.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

CLIENT_NOT_AUTHORIZED

Exception code for CLIENT_NOT_AUTHORIZED.

This code is used when an client performs an action that requires the user to be logged in and the user has not logged in yet.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

DATA_ACCESS_RESOURCE_FAILURE

Exception code for DATA_ACCESS_RESOURCE_FAILURE.

This code is used when a database exception received an error accessing storage devices.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

DATA_INTEGRITY_VIOLATION

Exception code for DATA_INTEGRITY_VIOLATION.

This code is used when a database exception is recognized as an integrity exception (like constraint violation).

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

DEADLOCK

Exception code for DEADLOCK.

This code is used when a deadlock is detected by the database.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

DELETE_NOT_GRANTED

Exception code for DELETE_NOT_GRANTED.

This code is used when a record deletion was rejected by a pre-delete Servoy trigger.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

EXECUTE_PROGRAM_FAILED

Exception code for EXECUTE_PROGRAM_FAILED.

This code is used when an external program was not executed correctly.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

INCORRECT_LOGIN

Exception code for INCORRECT_LOGIN.

This code is used when the user enters invalid credentials.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

INVALID_INPUT

Exception code for INVALID_INPUT.

This code is used when the user enters data that could not be validated.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

INVALID_RESULTSET_ACCESS

Exception code for INVALID_RESULTSET_ACCESS.

This code is used when a data is requested that is not selected in the sql.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

MAINTENANCE_MODE

Exception code for MAINTENANCE_MODE.

This code is used when a client could not be registered with the server because the server is in maintenance mode.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

NO_ACCESS

Exception code for NO_ACCESS.

This code is used when a user wants to view data and this is disallowed by security settings.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

NO_CREATE_ACCESS

Exception code for NO_CREATE_ACCESS.

This code is used when a user wants to create new records and this is disallowed by security settings.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

NO_DELETE_ACCESS

Exception code for NO_DELETE_ACCESS.

This code is used when a user wants to delete data and this is disallowed by security settings.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

NO_LICENSE

Exception code for NO_LICENSE.

This code is used when a client could not be registered with the server because of license limitations.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

NO_MODIFY_ACCESS

Exception code for NO_MODIFY_ACCESS.

This code is used when a user wants to update data and this is disallowed by security settings.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

NO_PARENT_DELETE_WITH_RELATED_RECORDS

Exception code for NO_PARENT_DELETE_WITH_RELATED_RECORDS.

This code is used when a record could not be deleted because a non-empty relation exists for the record that does not allow parent delete when having related records.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
        application.output("is a ServoyException")
        application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace() trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace() trace = servoyException.getStackTrace();
        if (servoyException.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)
                {
                    /** @type {DataException} */
                    var dataException = record.exception;
                    application.output('SQL: '+dataException.getSQL())
                    application.output('SQLState: '+dataException.getSQLState())
                    application.output('VendorErrorCode: '+dataException.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
```

NO_RELATED_CREATE_ACCESS

Exception code for NO_RELATED_CREATE_ACCESS.

This code is used when a user wants to create new related records and this is disallowed by security settings.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

PERMISSION_DENIED

Exception code for PERMISSION_DENIED.

This code is used when a database exception is recognized as a authorization error.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

RECORD_LOCKED

Exception code for RECORD_LOCKED.

This code is used when a record could not be updated or deleted because it is locked by another client.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

RECORD_VALIDATION_FAILED

Exception code for RECORD_VALIDATION_FAILED.

This code is used when a record update/insert was rejected by a pre-update/insert Servoy trigger.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

SAVE_FAILED

Exception code for SAVE_FAILED.

This code is used when a javascript exception occurred during saving data to the database.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

UNEXPECTED_UPDATE_COUNT

Exception code for UNEXPECTED_UPDATE_COUNT.

This code is used when a data could not be deleted or updated when expected (for example when a record was deleted outside Servoy and a Servoy client wants to update the record).

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace() trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace() trace = servoyException.getStackTrace();
        if (servoyException.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)
                {
                    /** @type {DataException} */
                    var dataException = record.exception;
                    application.output('SQL: '+dataException.getSQL())
                    application.output('SQLState: '+dataException.getSQLState())
                    application.output('VendorErrorCode: '+dataException.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
```

UNKNOWN_DATABASE_EXCEPTION

Exception code for UNKNOWN_DATABASE_EXCEPTION.

This code is used when an unrecognized database exception has occurred.

Returns

Number

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

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
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```

[getStackTrace](#)

String [getStackTrace\(\)](#)

Returns the stack trace for this ServoyException.

Returns

String – the string stack trace for this ServoyException.

Sample

```
//this sample script should be attached to onError method handler in the solution settings
application.output('Exception Object: '+ex)
application.output('MSG: '+ex.getMessage())
if (ex instanceof ServoyException)
{
    /** @type {ServoyException} */
    var servoyException = ex;
    application.output("is a ServoyException")
    application.output("Errorcode: "+servoyException.getErrorCode())
    var trace = "";
    if (ex.getScriptStackTrace) trace = servoyException.getScriptStackTrace();
    else if (servoyException.getStackTrace) trace = servoyException.getStackTrace();
    if (servoyException.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)
            {
                /** @type {DataException} */
                var dataException = record.exception;
                application.output('SQL: '+dataException.getSQL())
                application.output('SQLState: '+dataException.getSQLState())
                application.output('VendorErrorCode: '+dataException.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
```