GemStone®

GemConnect™ Release Notes

Version 2.3

April 2014

INTELLECTUAL PROPERTY OWNERSHIP

This documentation is furnished for informational use only and is subject to change without notice. GemTalk Systems, LLC, assumes no responsibility or liability for any errors or inaccuracies that may appear in this documentation.

This documentation, or any part of it, may not be reproduced, displayed, photocopied, transmitted, or otherwise copied in any form or by any means now known or later developed, such as electronic, optical, or mechanical means, without express written authorization from GemTalk Systems.

Warning: This computer program and its documentation are protected by copyright law and international treaties. Any unauthorized copying or distribution of this program, its documentation, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted under the maximum extent possible under the law.

The software installed in accordance with this documentation is copyrighted and licensed by GemTalk Systems under separate license agreement. This software may only be used pursuant to the terms and conditions of such license agreement. Any other use may be a violation of law.

Use, duplication, or disclosure by the Government is subject to restrictions set forth in the Commercial Software - Restricted Rights clause at 52.227-19 of the Federal Acquisitions Regulations (48 CFR 52.227-19) except that the government agency shall not have the right to disclose this software to support service contractors or their subcontractors without the prior written consent of GemTalk Systems.

This software is provided by GemTalk Systems, LLC and contributors “as is” and any expressed or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall GemTalk Systems, LLC or any contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

COPYRIGHTS

This software product, its documentation, and its user interface © 1986-2014 GemTalk Systems, LLC. All rights reserved by GemTalk Systems.

PATENTS

GemStone software is covered by U.S. Patent Number 6,256,637 “Transactional virtual machine architecture”, Patent Number 6,360,219 “Object queues with concurrent updating”, Patent Number 6,567,905 “Generational garbage collector with persistent object cache”, and Patent Number 6,681,226 “Selective pessimistic locking for a concurrently updateable database”. GemStone software may also be covered by one or more pending United States patent applications.

TRADEMARKS

VMware is a registered trademark or trademark of VMware, Inc. in the United States and/or other jurisdictions.

GemStone, **GemBuilder**, **GemConnect**, and the GemStone logos are trademarks or registered trademarks of GemTalk Systems, LLC., or of VMware, Inc., previously of GemStone Systems, Inc., in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Sun, **Sun Microsystems**, and **Solaris** are trademarks or registered trademarks of Oracle and/or its affiliates. **SPARC** is a registered trademark of SPARC International, Inc.

HP, **HP Integrity**, and **HP-UX** are registered trademarks of Hewlett Packard Company.

Intel, Pentium, and **Itanium** are registered trademarks of Intel Corporation in the United States and other countries.

Microsoft, **MS**, **Windows, Windows XP, Windows 2003**, **Windows 7**, **Windows Vista** and **Windows 2008** are registered trademarks of Microsoft Corporation in the United States and other countries.

Linux is a registered trademark of Linus Torvalds and others.

Red Hat and all Red Hat-based trademarks and logos are trademarks or registered trademarks of Red Hat, Inc. in the United States and other countries.

SUSE is a registered trademark of Novell, Inc. in the United States and other countries.

AIX, **POWER5**, **POWER6**, and **POWER7** are trademarks or registered trademarks of International Business Machines Corporation.

Apple, **Mac**, **Mac OS, Macintosh**, and **Snow Leopard** are trademarks of Apple Inc., in the United States and other countries.

Other company or product names mentioned herein may be trademarks or registered trademarks of their respective owners. Trademark specifications are subject to change without notice. GemTalk Systems cannot attest to the accuracy of all trademark information. Use of a term in this documentation should not be regarded as affecting the validity of any trademark or service mark.

GemTalk Systems
15220 NW Greenbrier Parkway
Suite 240
Beaverton, OR 97006

Preface

About the Documentation

These release notes describe the changes in the GemConnect® version 2.3 release. We recommend that everyone using GemConnect read these release notes before beginning installation or development.

For information on installing or upgrading to this version of GemConnect, please refer to the *GemConnect Installation Guide* for version 2.3.

These documents are available on the GemTalk support website, as described below.

Terminology Conventions

The term "GemStone" is used to refer to the server products GemStone/S 64 Bit and GemStone/S, and the GemStone family of products; the GemStone Smalltalk programming language; and may also be used to refer to the company, now GemTalk Systems, previously GemStone Systems, Inc. and a division of VMware, Inc.

Technical Support

Support Website

http://gemtalksystems.com/techsupport

GemTalk's Technical Support website provides a variety of resources to help you use GemTalk products:

- **Documentation** for released versions of GemTalk products, in PDF form.
- **Downloads**, including current and recent versions of GemTalk products.
- **Bugnotes**, identifying performance issues or error conditions that you may encounter when using a GemTalk product.

- **TechTips**, providing information and instructions that are not in the documentation.
- **Compatibility matrices**, listing supported platforms for GemTalk product versions.

This material is updated regularly; we recommend checking this site on a regular basis.

Help Requests

You may need to contact Technical Support directly, if your questions are not answered in the documentation or by other material on the Technical Support site. Technical Support is available to customers with current support contracts.

Requests for technical assistance may be submitted online, by email, or by telephone. We recommend you use telephone contact only for more serious requests that require immediate evaluation, such as a production system down. The support website is the preferred way to contact Technical Support.

Website: http://techsupport.gemtalksystems.com

Email: techsupport@gemtalksystems.com

Telephone: (800) 243-4772 or (503) 766-4702

When submitting a request, please include the following information:

- Your name and company name.
- The versions of GemConnect, Oracle, and of all related GemTalk products, and of any other related products.
- The operating system and version you are using.
- A description of the problem or request.
- Exact error message(s) received, if any, including log files if appropriate.

Technical Support is available from 8am to 5pm Pacific Time, Monday through Friday, excluding GemTalk holidays.

24x7 Emergency Technical Support

GemTalk offers, at an additional charge, 24x7 emergency technical support. This support entitles customers to contact us 24 hours a day, 7 days a week, 365 days a year, for issues impacting a production system. For more details, contact GemTalk Support Renewals.

Training and Consulting

GemTalk Professional Services provide consulting to help you succeed with GemStone products. Training for GemStone/S is available at your location, and training courses are offered periodically at our offices in Beaverton, Oregon. Contact GemTalk Professional Services for more details or to obtain consulting services.

Contents

Chapter
1

GemConnect 2.3 Release Notes

GemConnect version 2.3 is a new release of the GemConnect product, providing support for the GemStone/S 64 Bit v3.2 server, and introducing a number of new features.

These release notes provide changes between the previous version of GemConnect, version 2.2.3, and version 2.3. If you are upgrading from a version prior to 2.2.3, please review the release notes for each intermediate release to see the full set of changes.

This version of GemConnect supports only GemStone/S 64 Bit v3.2 and later.

For instructions on installing and upgrading, see the *GemConnect Installation Guide* for version 2.3.

Server keyfiles and GemConnect

With the release of GemStone/S 64 Bit v3.2 and GemConnect version 2.3, authorization for using GemConnect is now provided by the GS/64 keyfile. You will need to have a keyfile that provides authorization for use with GemConnect in order to use GemConnect.

To obtain a new keyfile for GemStone/S v3.2 and GemConnect (and GemBuilder for Java, if you are using that product), write to keyfiles@gemtalksystem.com. In your request, include the specific request for GemConnect, as well as your license information, platform, and any updates to contact information. It may be helpful to include your current keyfile with the request.

Contact GemTalk Technical Support if you have issues or questions.

Beta version with Kerberos Support available

If you are interested in using Kerberos to authenticate Oracle logins, contact GemTalk Technical Support for a version of the GemConnect library with Kerberos support, for evaluation and testing.

Supported Platforms

GemConnect version 2.3 is tested and supported on the following GemStone server product versions and platforms:

GemStone/S 64 Bit 3.2
‣ Solaris 10 and 11 on SPARC ‣ Solaris 10 on x86 ‣ AIX 6.1 and AIX 7.1 ‣ Ubuntu 10.04 on x86 ‣ Red Hat Linux ES 6.1 on x86 ‣ Mac OSX 10.6.4 with Darwin 10.4.0 on x86

Changes in this release

National Character Support

Oracle database may use one of two character representations, which are configured when the DB is created:

- The "standard" character set; this is usually used for internal string purposes like the names of tables and fields within the database, or for all strings when ASCII characters are sufficient. It is usually configured for an 8-bit representation, such as ASCII or UTF8.
- The "national" character set; used for representing larger character sets, like latin characters with diacritic marks or non-latin language sets. This is often configured for UTF16.

Previously, GemConnect had a single instvar on the GsOracleConnection, #charConversion. This differentiated between String and UTF8, but did not distinguish between standard and national character sets, and did not allow any control over the specific string type that was returned.

Version 2.3 includes significant enhancements in the handling for character data.

- the encoding for the standard and national character sets of the oracle database can be specified in the GsOracleConnection
- the specific string class for returned values can be specified, and whether to error if out of range characters would prevent returning that type of String
- Errors in UTF-8 data can be silently handled, rather than raising an error.
- The full set of unicode code points is supported, and you can return instances of QuadByteString or Unicode32.
- National character set datatypes for Oracle fields are now supported: NCHAR, NCVARCHAR2, and NCLOB.

Changes in GsOracleConnection instance variables

There are five added GsOracleConnection instVars:

`charSet`	The standard character set type used in the Oracle DB. Applies for CHAR, VARCHAR2, LONG, RAW, and CLOB fields
`ncharSet`	The national character set type used in the Oracle DB. Applies for NCHAR, NVARCHAR2, and NCLOB fields
`charGS`	Controls how standard character set fields are returned to GemStone. Applies for fields that are CHAR, VARCHAR2, LONG, RAW, and CLOB.
`ncharGS`	Controls how national character set fields are returned to GemStone. Applies to fields that are NCHAR, NVARCHAR2, and NCLOB.
`utfErr`	Controls whether an error should be triggered on UTF conversion, if there is a out of range character.

The instance variable #charConversion is obsolete.

charSet and ncharSet

The new instance variables `charSet` and `ncharSet` are used to specify the expected character set type for the Oracle database. These are determined based on the configuration of your Oracle database.

Possible values are:

#CHR8	A fixed-width character set encoded in 8-bit fields. The default for `charSet`. This would be the setting when Oracle uses character set US7ASCII, WE8MSWIN1252. or WE8ISO8859P1.
#CHR16	A fixed-width character set encoded in 16-bit fields. The default for `ncharSet`. This would be the setting when Oracle uses character set JA16SJIS.
#UTF8	UTF8 variable-width character set (8-bit fields). This would be the setting when Oracle uses character set AL32UTF8.
#UTF16	UTF16 variable-width character set (16-bit fields). This would be the setting when Oracle uses character set AL16UTF16.
nil	Use default behavior: #CHR8 for `charSet`, #CHR16 for `ncharSet`

In the US/Europe, the standard character set is usually 8-bit based (for example US7ASCII, WE8MSWIN1252, WE8ISO8859P1, or AL32UTF8), while the national character set is usually 16-bit based (for example JA16SJIS or AL16UTF16). You can check the character sets used in your Oracle DB by executing the following SQL from sqlplus:

```
sql> select * from nls_database_parameters where parameter like
'NLS%CHARACTERSET';
```

For example, executing this command against an Oracle DB that uses UTF8 for the standard character set and UTF-16 for the national character set produces the following results, which you would use to set the charSet/ncharSet:

Oracle Parameter	result	charSet/ncharSet value
`NLS_CHARACTERSET`	`AL32UTF8`	charSet: #UTF8
`NLS_NCHAR_CHARACTERSET`	`AL16UTF16`	ncharSet: #UTF16

Incorrect setting of these two instVars can result in incorrect mapping of characters in strings as they are read/written to/from Oracle. However, you may intentionally set the GemConnect instance variable to not match Oracle, if desired. For example, if you want to get back un-decoded UTF8 strings, setting the charSet to #CHR8 would disable UTF conversion.

charGS and ncharGS

The new instance variables charGS and ncharGS control the type of GemStone string that is returned from Oracle, and whether to error if the oracle data includes characters that are out of range for the GemStone string type. charGS applies for standard character set fields, ncharGS applies for national character set fields.

- Standard Strings are String, DoubleByteString and QuadByteString.
- Unicode Strings are Unicode7, Unicode16 and Unicode32. These are similar to standard strings, but rely on the ICU libraries to perform language-specific collation and other operations.
- ByteArray and Uft8 are arrays of bytes, convertible to and from Strings.

Possible values for charGS and ncharGS are:

`#STR` or nil	return a String, DoubleByteString, or QuadByteString, as appropriate. This is the default.
`#BYTE`	ignore charSet/ncharSet, and return the raw data in a ByteArray.
`#STX`	return only instances of String. Raise an error if a character is out of the String range.
`#DBX`	return only instances of DoubleByteString. Raise an error if a character is larger than the DoubleByteString range.
`#QBX`	return only instances of QuadByteStrings.
`#DBS`	Return a String or DoubleByteString as appropriate. Raise an error if a character is out of the DoubleByteString range.
`#QBS`	Return a String, DoubleByteString, or QuadByteString as appropriate. This is the same as #STR.
`#UNC`	Return a Unicode7, Unicode16, or Unicode32, as appropriate
`#UC7`	Return only instances of Unicode7. Raise an error if a character is out of the Unicode7 range.
`#UC16`	Return only instances of Unicode16. Raise an error if a character is out of the Unicode16 range.

`#UC32`	Return only instances of Unicode32.
`#UTF8`	Return only instances of Utf8.
`#UTF16`	return only Utf16 strings (not yet supported)

Write Operations

For write operations to Oracle, any GemStone String class may be used. The setting for `charSet` and `ncharSet` will be used to make the appropriate conversion.

You may also write ByteArrays. This disables conversion; the bytes are written directly. If the bytes represent UTF8, it is the developer's responsibility ensure that the ByteArray represents a valid UTF8 String . Instances of Uft8, a subclass of ByteArray, are already encoded as UTF8.

utfErr

The new instance variable `utfErr` controls how GemConnect will deal with conversion problems of UTF8 or UTF16 strings.

TRUE or nil	Raise an error on any conversion problem. (the default)
FALSE	Attempt to silently handle UTF conversion problems. If converting a character that is invalid for Unicode, will replace it with either the standard Unicode Replacement Character (0xFFFD), or in a single byte string, with a question mark.

BLOB/CLOB support for INSERT/UPDATE operations (bug 42343)

When BLOBS/CLOBS were introduced in GemConnect 2.1, they were accessed via SQL SELECT statements and could be modified using SQL SELECT FOR UPDATE statements. But there was no easy way to do an SQL INSERT or UPDATE on a table containing a BLOB/CLOB.

You can now use bound INSERT/UPDATE SQL statements and use #openInsertCursorOn:* and #openUpdateCursorOn:* on tables containing BLOBS/CLOBS. For a table field containing a BLOB, a ByteArray can be used as a value, while for CLOBS, an appropriate String/DoubleByteString can be used.

Support for :bind variables in SELECT statements

In order to support SELECT queries with variables that are bound to values after query creation, a number of new GsOracleConnection methods have been added. These methods return an instance of a new class, **GsRdbSelectStream**. You can then bind values to the variables on this instance, and stream over the query results.

These methods are of the form:

```
GsOracleConnection >>
   openSelectCursorOn: sql
   tupleClass: tupleClass
   columnMapping: colMap
   bindInfo: bindInfo
```

tupleClass and *colMap* are used as in all other `open*CursorOn:*` methods. They may be nil; convenience methods are provided so these arguments do not always need to be included.

bindInfo is an array used to specify the bindings for the provided select statement. *bindInfo* is an array of 2 or more elements:

- The first element of this array is the name of the table this operation is performed against, or nil.
- The remainder of the *bindInfo* array is composed of 2-element arrays containing binding data for individual bindings. Each of these sub-arrays contains the name of the bind variable, starting with ":", and the type, which maybe Integer, Float, String, or DateTime.

For example, if the SQL statement is

```
'SELECT * FROM MYTABLE KEY < :KEYVAL AND SIZE > :SIZEVAL'
```

bindInfo could consist of:

```
#('MYTABLE' #(':KEYVAL' Integer) #(':SIZEVAL' Integer))
```

To avoid the need to construct the bindInfo array, variations of the `openSelectCursorOn:*` method are also provided, which include keywords for passing in the variable and type, for one or two :bind variables.

The following methods are provided:

```
openSelectCursorOn:bindInfo:
openSelectCursorOn:tupleClass:bindInfo:
openSelectCursorOn:columnMapping:bindInfo:
openSelectCursorOn:tupleClass:columnMapping:bindInfo:
```

For one bind variable:

```
openSelectCursorOn:bind:type:
openSelectCursorOn:tupleClass:bind:type:
openSelectCursorOn:columnMapping:bind:type:
openSelectCursorOn:tupleClass:columnMapping:bind:type:
```

For two bind variables:

```
openSelectCursorOn:bind:type:bind:type:
openSelectCursorOn:tupleClass:bind:type:bind:type:
openSelectCursorOn:columnMapping:bind:type:bind:type:
openSelectCursorOn:tupleClass:columnMapping:bind:type:bind:type:
```

GsRdbSelectStream

Results from `GsOracleConnection>>openSelectCursorOn:*` are returned as an instance of GsRdbSelectStream.

Using the method `bindVars:` with an array of values, you can specify the specific values, and use stream protocol to read the results from the GsRdbSelectStream, just as you would for a normal select statement executed in a #openCursorOn:* call.

The number of arguments in the array specified by `bindVars:` must not be less than the number of :bind variable arrays that were specified in *bindInfo*. Failure to bind all variables will result in a #selectUnboundError. You must also order the values correctly according to the *bindInfo* array.

`nextPut:` is also supported and behaves similarly to `bindVars:`; this provides similar behavior to existing methods to perform database operations.

Example

The following example illustrates how you might use openSelectCursorOn: and a GsRdbSelectStream.

```
sql := 'SELECT * FROM MYTABLE WHERE KEY < :KEYVAL AND SIZE >
:SIZEVAL'.
bindInfo := Array new.
bindInfo
   with: 'MYTABLE'
   with: (Array with: ':KEYVAL' with: Integer)
   with: (Array with: ':SIZEVAL' with: Integer).
stream := myConnection openSelectCursorOn: sql bindInfo: bindInfo.
"following #bindVars: binds 12 to :KEYVAL and 100 do :SIZEVAL"
stream bindVars: (Array with: 12 with: 100).
result1 := stream next. "Returns 1st result for bound select"
result2 := stream next. "Returns 2nd result for bound select"
...
"following #bindVars: binds 20 to :KEYVAL and 50 do :SIZEVAL"
stream bindVars: (Array with: 20 with: 50).
result3 := stream next.  "Returns 1st result for new bound select"
results := stream upToEnd. "Returns remaining results"
```

Support for :bind variables in BLOCK statements

A method has also been added to support :bind variables in block statements. This method returns an instance of **GsRdbBlockStream**.

The following method has been added:

```
GsOracleConnection >>
   openBlockCursorOn: sql
   bindInfo: bindInfo
```

sql can be any SQL block statement that starts with BEGIN or DECLARE.

bindInfo is an array used to specify the bindings for the provided block statement; it is similar to that used for select statements, but takes an additional argument. *bindInfo* is an array of three elements:

- The first element of this array is nil. This is to maintain consistency with the structure of *bindInfo* arrays elsewhere, in which this slot references the name of the table.

- The remainder of the *bindInfo* array is composed of 2, 3, or 4-element arrays containing binding data for individual bindings. Each of these sub-arrays contains the following elements. The first two are required.

element 1:	Name of the bind variable, starting with ":".
element 2:	Type of the bind variable: one of Integer/Float/String/DateTime
element 3:	(optional): Direction for bind variable: #IN #OUT or #INOUT Default, if this is omitted, is #IN
element 4:	(optional, requires that element 3 be included) For String bind variables, the maximum size of string allowed. Strings exceeding this size will be silently truncated. Default is the connection's textLimit (which defaults to 65532).

GsRdbBlockStream

Results from `GsOracleConnection>>openBlockCursorOn:bindInfo:` are returned as an instance of GsRdbBlockStream.

Using the method bindVars: with an array of arguments, you can specify the specific values, and use stream protocol to read the results from the GsRdbBlockStream.

The number of arguments in the array specified by `bindVars:` must not be less than the number of :bind variable arrays that were specified in *bindInfo*. #OUT variables are ignored; you may specify nil for these variables. Failure to bind all variables will result in a #selectUnboundError. You must also order the values correctly according to the *bindInfo* array.

`nextPut:` is also supported and behaves similarly to `bindVars:`; this provides similar behavior to existing methods to perform database operations.

Results are returned as an Array, with one element for each bind variable. #IN variables are returned without change, #INOUT and #OUT will contain the output value for that variable.

Example

The following example executes a simple block that accepts a single integer and returns as a result that is the value doubled

```
sql := '
BEGIN
:OUTPUT := :INPUT + :INPUT;
END;
'.
bindInfo := Array
   with: nil
   with: (Array with: ':INPUT' with: Integer with: #IN)
   with: (Array with: ':OUTPUT' with: Integer with: #OUT).
stream := myConnection openBlockCursorOn: sql bindInfo: bindInfo.
results := stream bindVars: (Array with: 42 with: nil).
```

Results will be the array #[42 84].

Errors are now GsRdbError

A new class has been added, GsRdbError, to allow GemConnect-specific errors to be returned as an ANSI rather than legacy error.

GsRdbError arguments are accessed by the following accessor methods:

```
connection
stream
sql
rdbErrDetails
```

The ordering of arguments in gsArgs has been standardized and follows this order (connection, stream, SQL command and details.

Details may be a string, or an array of further details.

#queryError, #invalidSql, #oracleError

For errors of these types, the details will be an array containing the following elements. While Oracle will usually return one error, it can also return a collection of multiple errors. In this case the array will be larger, as each error is added to the Array.

1 - Oracle Call Interface (OCI) return code (usually -1)
2 - Number of errors (usually 1, but multiple errors can occur)
3 - Oracle error code
4 - Oracle error message
5 - GemStone object causing problems
[Repeat slots 3-5 for each additional error]
N - (last element) Internal diagnostic message from GemStone

#typeConversionError

For errors of this type, the details will be an array containing the following elements:

1: Row object
2: Selector (optional)
3: Column number
4: The Object causing conversion error
5: Comment

Changes in Errors

Over time, the use of GemConnect errors 6 (#oracleError), 7 (#queryError), and 21 (#invalidSql) have become inconsistent. The following are the new standards for these errors:

21 #invalidSql
The SQL statement is invalid; it is syntactically incorrect.
Generated during the Oracle prepare statement phase.

7 #queryError
An error was encountered while performing a query.
Generated during the Oracle execute statement phase.

6 #oracleError
All other errors generated from Oracle

Customers should review their application exception handlers and modify them as appropriate.

New errors

29 #bindingSetupError
Error in binding specification, connection: <c> query: <sql>

30 #bindingValueError
Error in binding value assignment, connection: <c> stream: <s> details: <msg>

31 #selectUnboundError
Attempting to read from a select cursor that has not yet had values assigned to its bindings, connection: <c> stream: <s>

Bugs Fixed

Reads of CLOB/NCLOB data may skip part of the object

There were cases in which reading CLOB/NCLOB data using OCILobRead() was not performed correctly, resulting in stream advance without returning all expected bytes. (#43754)

Commit work on Solaris on x86 may fail Oracle type check

On Solaris on x86, COMMIT WORK may have resulted in an error. (#43903)

Optimization to avoid unnecessary version check

GemConnect notification tracking performs a version check, which was introduced to avoid problems with older versions of the GemStone/S Server that did not support notification. Since these older versions are no longer supported by current versions of GemConnect, this check is unnecessary and had been removed. (#41572)