GemStone®

GemStone/S 64 Bit Release Notes

Version 2.4

September 2009

INTELLECTUAL PROPERTY OWNERSHIP

This documentation is furnished for informational use only and is subject to change without notice. GemStone Systems, Inc. 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 GemStone Systems, Inc.

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 GemStone Systems, Inc. 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 GemStone Systems, Inc.

This software is provided by GemStone Systems, Inc. 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 GemStone Systems, Inc. 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

PATENTS

GemStone is covered by U.S. Patent Number 6,256,637 “Transactional virtual machine architecture”, Patent Number 6,360,219 “Object queues with concurrent updating”, and Patent Number 6,567,905 “Generational Garbage Collector”. GemStone may also be covered by one or more pending United States patent applications.

TRADEMARKS

GemStone, **GemBuilder**, **GemConnect**, and the GemStone logos are trademarks or registered trademarks 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**, **Solaris**, and **SunOS** are trademarks or registered trademarks of Sun Microsystems, Inc. All **SPARC** trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. **SPARCstation** is licensed exclusively to Sun Microsystems, Inc. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

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

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

Microsoft, **MS**, **Windows, Windows XP, Windows 2000**, **Windows 2003**, and **Windows Vista** 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.

AIX and **POWER4, POWER5**, and **POWER6** are trademarks or registered trademarks of International Business Machines Corporation.

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. All terms mentioned in this documentation that are known to be trademarks or service marks have been appropriately capitalized to the best of our knowledge; however, GemStone 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.

GemStone Systems, Inc.
1260 NW Waterhouse Avenue, Suite 200
Beaverton, OR 97006

Preface

About This Documentation

These release notes describe changes in the GemStone/S 64 Bit version 2.4 release. We recommend that everyone migrating to this version read these release notes before beginning installation, testing or development.

For information on installing or upgrading to this version of GemStone/S 64 Bit, please refer to the *GemStone/S 64 Bit Installation Guide.*

These documents are also available on the GemStone customer website, as described below.

Terminology Conventions

This document uses the following terminology:

The term "GemStone" is used to refer both to the product, GemStone/S 64 Bit, or previous GemStone/S server products; and to the company, GemStone Systems, Inc.

Technical Support

GemStone provides several sources for product information and support. The product-specific manuals provide extensive documentation, and should be your first source of information.

GemStone Web Site: http://support.gemstone.com

GemStone's Technical Support website provides a variety of resources to help you use GemStone products. Use of this site requires an account, but registration is free of charge. To get an account, just complete the Registration Form, found in the same location. You'll be able to access the site as soon as you submit the web form.

The following types of information are provided at this web site:

Documentation for GemStone/S 64 Bit is provided in PDF format. This is the same documentation that is included with your GemStone/S 64 Bit product.

Release Notes and **Install Guides** for your product software are provided in PDF format in the Documentation section.

Downloads and **Patches** provide code fixes and enhancements that have been developed after product release, and past and current versions of GemBuilder for Smalltalk.

Bugnotes, in the Learning Center section, identify performance issues or error conditions that you may encounter when using a GemStone product. A bugnote describes the cause of the condition, and, when possible, provides an alternative means of accomplishing the task. In addition, bugnotes identify whether or not a fix is available, either by upgrading to another version of the product, or by applying a patch. Bugnotes are updated regularly.

TechTips, also in the Learning Center section, provide information and instructions for topics that usually relate to more effective or efficient use of GemStone products.

Community Links provide customer forums for discussion of GemStone product issues.

Technical information on the GemStone Web site is reviewed and updated regularly. We recommend that you check this site on a regular basis to obtain the latest technical information for GemStone products.

Help Requests

You may need to contact Technical Support directly for the following reasons:

- Your technical question is not answered in the documentation.
- You receive an error message that directs you to contact GemStone Technical Support.
- You want to report a bug.
- You want to submit a feature request.

Questions concerning product availability, pricing, keyfiles, or future features should be directed to your GemStone account manager.

When contacting GemStone Technical Support, please be prepared to provide the following information:

- Your name, company name, and GemStone/S license number
- The GemStone product and version you are using
- The hardware platform and operating system you are using
- A description of the problem or request
- Exact error message(s) received, if any

Your GemStone support agreement may identify specific individuals who are responsible for submitting all support requests to GemStone. If so, please submit your information through those individuals. All responses will be sent to authorized contacts only.

For non-emergency requests, the support website is the preferred way to contact Technical Support. Only designated support contacts may submit help requests via the support website. If you are a designated support contact for your company, or the designated contacts have changed, please contact us to update the appropriate user accounts.

Website: http://techsupport.gemstone.com

Email: support@gemstone.com

Telephone: (800) 243-4772 or (503) 533-3503

Requests for technical assistance may be submitted online, or 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 that is non-operational. In these cases, please also submit your request via the web or email, including pertinent details such error messages and relevant log files.

If you are reporting an emergency by telephone, select the option to transfer your call to the technical support administrator, who will take down your customer information and immediately contact an engineer.

Non-emergency requests received by telephone will be placed in the normal support queue for evaluation and response.

24x7 Emergency Technical Support

GemStone 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, if they encounter problems that cause their production application to go down, or that have the potential to bring their production application down. For more details, contact your GemStone account manager.

Training and Consulting

Consulting and training for all GemStone products are available through GemStone's Professional Services organization.

- Training courses are offered periodically at GemStone's offices in Beaverton, Oregon, or you can arrange for onsite training at your desired location.
- Customized consulting services can help you make the best use of GemStone products in your business environment.

Contact your GemStone account representative for more details or to obtain consulting services.

Contents

Chapter 1. GemStone/S 64 Bit 2.4 Release Notes

Chapter
1

GemStone/S 64 Bit 2.4 Release Notes

Overview

GemStone/S 64 Bit 2.4 is a new version of the GemStone/S 64 Bit object server. This release provides new features and performance improvements, and fixes a number of bugs; we recommend everyone using GemStone/S 64 Bit upgrade to this new version.

These release notes provide details of the changes between the previous version of GemStone/S 64 Bit, version 2.3.1.6, and version 2.4. If you are upgrading from a version prior to 2.3.1.6, please also review the release notes for each intermediate release to see the full set of changes.

Note that changes in internal indexing structures between 2.2.x and 2.3 mean that indexes cannot be upgraded. If you are upgrading from a version earlier than 2.3 to version 2.3 or later, you must drop and rebuild all indexes in your system.

For details about installing GemStone/S 64 Bit 2.4 or upgrading from earlier versions of GemStone/S 64 Bit or other GemStone server products, see the *GemStone/S 64 Bit Installation Guide* for version 2.4.

There are changes to the OS configuration requirements; additional semaphores are required. See “Additional Semaphores required” on page 2, and the System Requirements section of Chapter 1 of the Installation Guide.

This version is available for Solaris on Sparc, Solaris on x86, AIX, HP on PA-RISC and HP on Itanium, and Linux.

Technical Support

GemStone Technical Support has a new Help Request system. Customers with support agreements are invited to submit Help Requests to

```
http://techsupport.gemstone.com
```

Changes and New Features

Additional Semaphores required

To fix bug #40096, “Risk of Stone hang or crash on Gem SIGTERM“, an additional semaphore has been added to each GemStone process; the number of semaphores that GemStone requires has doubled in this release. On platforms for which the number of semaphores is configurable, the operating system configuration must be reviewed and the semaphore limit increased.

WARNING
Existing GemStone/S systems may require OS kernel adjustments in order to operate after upgrading to version 2.4.

Performance improvements and internal changes

Redesigned parallel transaction log recovery/restore

In order to more efficiently replay transaction logs, the restore process has been redesigned to use multiple threads running in parallel to read, analyze, and process the tranlog records. In addition, the GcGems now run during tranlog restore, to offload the reclaim work previously done by the Stone during tranlog restore. This greatly improves performance of tranlog replay for larger systems; testing results show a 5x performance improvement. This will allow warm standbys to keep up with tranlogs generated during periods of intense production system activity.

The interface for tranlog reply is unchanged, however there are some differences in system activity compared to restore under earlier versions.

While executing parallel restore from tranlog, unlike the older tranlog restore mechanism, the Reclaim GcGems are running.

AIO wait behavior

The behavior of aio_suspend() on Solaris caused problems with GemStone AIO requests; see “Performance problems due to AIO behavior” on page 20.

The error returns codes from aio_suspend were not always as documented, so to avoid problems when aio_suspend() is called on an AIO request that had completed, error return codes from aio_suspend() are no longer used.

If the return from the aio_suspend() function indicated AIO was not completed, then aio_error() is called to determine the actual status. aio_suspend() is only called a second time if aio_error returns EINPROGRESS. Calls to aio_suspend() may incorrectly suspend for the full timeout, so the timeout argument has been reduced.

Previously AIO requests for which multiple aio_suspend() calls returned EINPROGRESS for more than 30 seconds were cancelled. This timeout has been increased to 60 seconds.

Previously, if a AIO request did not complete within the timeout period, all outstanding AIO requests were cancelled. Now only the specific request is cancelled, and that request is retried using synchronous write.

A number of cache statistics have been added to track Aio behavior; these statistics begin with “StnAio”. See the cache statistics starting on page 19.

Reduced network round trips during login

The performance of login, in particular remote logins on systems with many extent files, has been improved. The number of network round trips required during login has been greatly reduced.

Previously, one call to the stone per extent was made during login; now, one call fetches all the extent file names. Also, socket code that determines the correct NRS for the extents is now cached, so it only is run once per session.

Previously, a network round trip was made to the remote netldi to resolve the stone's port for each extent; now, this information is cached following the first execution.

The number of round trips to the gem's page server to open the extent has been reduced from four to one.

Reduced round trips from System class >> currentSessionNames

This method now does one call to the stone per 256 sessions, reduced from two calls.

Reduced round trips from Repository >> fileSizeReport

The number of round trips to the stone required for this method is greatly reduced. This method now makes two or three total calls to the stone, rather than two calls per extent.

This improvement is implemented by a new feature in `Repository >> _extentStatistics:`. The methods `freeSpace` and `fileSize` also use this new feature and will have similar improvements.

Shared Page Cache Monitor

Internal structures have been redesigned to improve performance for the SPC monitor, and place an upper limit on the amount of CPU used for statistics collection.

New cache statistics have been added to allow monitoring for each of the four threads, providing an estimate of the CPU usage of each:

MainThreadTimeRunningMs
CrashedSlotThreadTimeRunningMs
CleanSlotThreadTimeRunningMs
StatsThreadTimeRunningMs

RestoreFromBackup adjusts AIO page server behavior

To keep the cache from overloading, the `restoreFromBackup` code now puts the AIO page servers in aggressive mode. The maximum I/O rate is set to 1000000, and data pages are preempted immediately after writing.

Improved handling of low free space conditions

In low free space conditions, the stone will now service the page manager while waiting for free space, to allow pages to be returned to make space available.

In addition, checkpoints during low free space conditions are now done every minute, rather than every three minutes.

Seaside/Glass

As in previous releases, the GemStone port of the open source Seaside web development framework is included.

Many improvements have been made, in particular to the tools for working in Seaside for GemStone. GemTools are now based on Pharo, a variant of Squeak.

This release also introduces Metacello, an open source version control and configuration management framework. Metacello works with Monticello to allow the definition of relationships and dependencies between Monticello packages and the local environment. This makes possible the definition of projects that can be smoothly loaded into multiple environments. For more information on Metacello, see:

`http://gemstonesoup.wordpress.com/2009/08/25/metacello-package-management-for-monticello/`

GemStone/S 64 Bit version 2.4 includes the Metacello-defined version of Seaside 2.8.3. Preliminary work has been done for the support of Seaside 2.9. This includes updates to GemStone server internals needed for Seaside 2.9 features, such as the implementation of PartialContinuations.

Further information about Seaside and Glass, see the website:

`http://seaside.gemstone.com/`

and blog:

`http://gemstonesoup.wordpress.com/`

Locking shared memory on Solaris

On Solaris 10, GemStone uses Intimate Shared Memory for the shared page cache. This implicitly locks the SPC into RAM, making setting the SHR_PAGE_CACHE_LOCKED unnecessary. When this is done, the following line is added to the SPC monitor log:

[Info]: Shared cache successfully created and locked using intimate shared memory.

New Feature for repository profiling

New classes have been added, GsObjectInventory and GsObjectInventoryEntry, to enable detailed profiling of a repository. These scan the repository similarly to listInstances.

`GsObjectInventory class >> profileRepository`

This method scans the entire repository and creates a profile by class. It returns a new instance of the receiver. The number of instances and the number of physical bytes occupied by each class in the repository with one or more instances are counted. This method stays in transaction for its entire duration, and holds the GcLock, so no garbage collection can be run while this method is executing.

`GsObjectInventory class >> profileGarbageFromFile:` *aFilename* `includeHiddenObjects:` *aBoolean*

This method scans the results of an FDC which were written to *aFilename*, and creates a profile based on the contents.

Parallel index creation

It is now possible to create multiple indexes on a given collection in parallel.

Under some conditions of collection size and number of indexes, this may result in significant performance improvements. However, other configurations may not show much improvement. Other factors, such as cache size, will also impact performance. To determine if there is a performance benefit for a particular situations, tests should be run.

To create parallel indexes, create instances of the new classes **EqualityIndexSpecification**, **IdentityIndexSpecification**, and **RcEqualityIndexSpecification**, and use the new method UnorderedCollection >> createParallelIndexes:.

For example,

```
| specArray |
specArray := #[
   (EqualityIndexSpecification path: 'name.last'
         lastElementClass: String),
   (IdentityIndexSpecification path: 'customerNumber'),
   (RcEqualityIndexSpecification path: 'status'
         lastElementClass: Symbol)
   ].
CustomerCollection createParallelIndexes: specArray.
```

Added methods

GsFile rename

The following methods have been added to allow files to be renamed on the client or server using GsFile:

GsFile class >> renameFileOnServer: *oldName* to: *newName*
GsFile class >> renameFile: *oldName* to: *newName*

Time since stone startup

New methods have been added to provide the elapsed time since the most recent stone startup:

System class >> secondsSinceStoneStarted
Return the SmallInteger number of seconds since the stone was started.

DateTime class >> whenStoneStarted
Return the instance of DateTime corresponding to the time the stone was started.

Disable almost out of memory warning

A method has been added to allow the almost out of memory signal to be disabled. This method may be used after the almost out of memory signal has previously been enabled.

System Class >> disableAlmostOutOfMemoryError

String formatting method

`CharacterCollection >> addLineWith:` *aCharacterCollection* `centeredToWidth:` *anInt*

Add a centered line of width anInt to the receiver.

Stream methods

For ANSI compliance, the following methods have been added:

`WriteStream class >> with:` *aCollection*

Create an instance of the receiver on *aCollection*, positioned at the end.

`PositionableStream >> setToEnd`

Set the receiver's position to the end

Methods to access session times

`System >> lastTransactionTimeForSessionWithId:` *aSessionId*

Returns a SmallInteger representing the time of the session's most recent beginTransaction, commitTransaction, or abortTransaction (from `System timeGmt`). Equivalent to executing: `(System descriptionOfSession: aSessionId) at: 5`

`System >> millisecondsSinceTransactionBoundary`

Answer a SmallInteger representing the number of milliseconds since this session updated its view of the repository by executing #abortTransaction, #beginTransaction, #continueTransaction, or #commitTransaction. Note that an attempt to commit which fails (due to transaction conflicts) is considered a transaction boundary since the session is shifted to the most recent view by the failed commit attempt.

`System >> millisecondsSinceLogin`

Answer a SmallInteger representing the number of milliseconds since this session logged in to the repository.

Class convenience methods

`Behavior >> theNonMetaClass`
`Metaclass >> theNonMetaClass`

When sent to a Class or to a Metaclass, always return the Class.

List instances in page order

A method has been added to perform a listInstances, returning the results in page order rather than ordered by OOP. This provides much better performance for operations that operate on many objects, such as instance migration.

`Repository >> listInstancesInPageOrder:` *anArrayOfClasses* `toFile:` *aString*

This method scans the repository for all committed instances of the classes contained in *anArrayOfClasses*, and stores the results into a new file specified in *aString*. *anArrayOfClasses* must be an instance of Array which contains between 1 and 2034 classes. This method runs entirely in a transaction.

All results of the scan (all instances of the classes in anArray) are stored in the file in page order (i.e., the same order as the objects are stored on disk) as of the time of the scan. Note that unlike other #listInstances methods, the file contains the complete list of all instances of all classes contained in anArray.

`Repository >> openPageOrderOopFile:` *aString*

Opens a binary page-ordered file of object IDs created by the Repository>>listInstancesInPageOrder:toFile: method. Returns a SmallInteger which is the ID of the opened file. This method acquires and retains the garbage collection lock.

`Repository >> readObjectsFromFileWithId:` *aSmallInt* `startingAt:` *startIndex* `upTo:` *endIndex* `into:` *anArray*

Reads and validates objects from a page-ordered file which was previously opened with the `Repository>>openPageOrderOopFile:` method. *aSmallInt* must be the result of the `Repository>>openPageOrderOopFile:` method. *startIndex* is the (1-based) index of the first object in the file to read, and *endIndex* is the index of the last object to read from the file. *anArray* must be an empty instance of Array when the resulting objects will be stored. Returns a SmallInteger which is the size of anArray. This will be the lesser of the number (*endIndex* - *startIndex* + 1) or, all remaining objects in the file.

The session must be in a transaction when this method is invoked.

It is possible that one or more object identifiers contained in the file are no longer valid due to garbage collection. Objects which are no longer valid have nil stored in their place in anArray. Objects that have been garbage collected and are in the free oop list or the dead not reclaimed set are considered to be invalid. For best performance, it is recommended that no more than 2034 objects be returned by any single invocation of this method. Returning a high number of objects will cause anArray to grow very large and may result in a fatal out-of-memory error.

`Repository >> closePageOrderOopFileWithId:` *aSmallInt*

Closes a file previously opened with the `Repository>>openPageOrderOopFile:` method and releases the garbage collection lock.

`Repository >> auditPageOrderOopFileWithId:` *aSmallInt*

Reads the page order oop file with the ID returned by the `Repository>>openPageOrderOopFile:` method and counts the number of objects in the file that are invalid. Invalid objects are objects that do not exist or are in the dead object set.

The session must be in a transaction when this method is invoked.

Returns an array containing 2 SmallIntegers, the total number of oops in the file, and the number of invalid oops in the file.

Extended Character Set has improved collation

GemStone/S 64 Bit v2.4 includes support for the Default Unicode Collation Element Table (DUCET), to improve collation and increase the options for customizing collation.

In GemStone, extended character set collation (ordering of characters for sort) is provided by the sequencing of characters within the CharacterDataTable structures. Previous releases ordered Characters according to the order in the Unicode data, which did not provide appropriate collation for some characters, such as ß.

The `CharTableUnicode510.dat` file provided with v2.4 is ordered according to the Default Unicode Collation Element Table (DUCET), as provided in the unicode sort file:

```
http://www.unicode.org/Public/UCA/latest/allkeys.txt
```

Correct collation depends on the language and usage, so this default collation will not be appropriate for all languages and all sorting conditions. For details, see the collation discussions in the Unicode Collation Algorithm:

`http://unicode.org/reports/tr10/`

The default collation using DUCET provides normal sorting order for English and most European languages; or collation can be customized for specific application needs.

The code that is provided as a extra filein to generate Character Data Tables from raw Unicode data, UnicodeData.gs, has been enhanced to allow the DUCET table to be used.

In addition to improved collation, v2.4 also includes the latest version of the Unicode data, version 5.1. GemStone documentation incorrectly indicated that previous releases included Unicode v5.1 data; in fact, previous releases included v4.1 data. To avoid confusion, the data files provided with GemStone releases have been renamed to include the version number.

The data file corresponding to the `CharTableUnicode.dat` file included in previous releases is now:

`$GEMSTONE/examples/CharTableUnicode410.dat`

The Unicode version 5.1, which was generated with the DUCET sort, is

`$GEMSTONE/examples/CharTableUnicode510.dat.`

The file `$GEMSTONE/examples/CharTableDefault.dat` has been renamed to `CharTableDefault.tab`. The change in the filename extension is to indicate that this file is in structured table format, unlike `CharTableUnicode*.dat`, which is in primitive table format.

For instructions on installing the Unicode Extended Character Set, or creating customized versions of the Character Data Tables, refer to the updated *System Administration Guide*, appendix F.

Unlike previous releases, installed Character Data Tables are now preserved across upgrade and conversion, including conversion from the 32-bit GemStone/S 6.x. product. The first entry in the table must be for the NUL character (ASCII zero).

ProfMonitor

ProfMonitorTree

Smalltalk code profiling output is now additionally displayable in a tree form. This provides another view of the profiling data, more similar to VisualWorks and Squeak output, and it may also clarify profiling of recursive code.

This feature is provided via a new class, **`ProfMonitorTree`**. The interface to this class is the same as ProfMonitor; the only difference in use is the additional data in the results.

Object creation profiling

While object creation profiling, as providing in earlier GemStone/S products, is not currently available, it is now possible to profile the creation of persistent objects.

When `traceObjectCreation: true` is specified, the profiling output includes the number and class of new persistent objects.

Ability to disable STN_GEM_TIMOUT

For sessions that need to be immune from the STN_GEM_TIMEOUT, it is now possible to disable to the timeout for a particular session, using the following new methods:

`System >> disableStoneGemTimeout`

Prevents the current session from being killed due lack of communication with the stone and / or inactivity. Use the `resumeStoneGemTimeout` method to re-enable the stone gem timeout. This method returns true if the action was successful or false if the stone gem timeout was previously disabled for this session.

`System >> resumeStoneGemTimeout`

Re-enables the stone gem timeout for the current session after it was previously disabled by the `disableStoneGemTimeout` method. This method returns true if the action was successful, or false if the stone gem timeout was already re-enabled or if it was not previously disabled for this session.

ANSI-compliant position option

GemStone's implementation of `Stream>>position` is not ANSI-compliant. the ANSI specification says that position "Returns the number of sequenced values in the receiver's past sequence values", while GemStone position returns the current 1-based index into the Stream.

To avoid breaking legacy code that depends on the current non-ANSI behavior, GemStone now provides multiple additional interfaces for position:

`Stream >> positionA`, `Stream >> positionA:`

Provide the **A**NSI-compliant behavior

`Stream >> positionL`, `Stream >> positionL:`

Provide the **L**egacy behavior

`Stream >> positionW`, `Stream >> positionW:`

Write a **w**arning message, including a stack showing the sender, to the Stone log, then provide the Legacy behavior.

For new repositories, by default in v2.4, the position and position: methods retain their legacy behavior. The default for new repositories will change to the ANSI behavior in a future release.

During upgrade, the upgrade examines a new variable in Globals, `#'PositionableStream_position'`, and installs the appropriate position methods accordingly. An existing value for this variable is not modified during upgrade or conversion, so upgraded repositories will keep the same behavior.

If you wish to make the ANSI-compliant version the default for your system, execute the following as SystemUser:

```
Globals at: #'PositionableStream_position' put: #'ANSI'.
PositionableStream compilePositionMethods.
```

Gem sessions now tolerate errors in configuration files

Syntax errors in the configuration file used by a gem process (RPC gem or linked session) now do not cause the initialization to fail. Now, the error is reported and the initialization continues. If the syntax error affects a specific parameter, this parameter will use the default value.

Note that this requires extra attention to the contents the log file, or header for a linked session, to ensure that the session configuration is set up as intended. This is particularly important after making any changes in the configuration files used by sessions.

Logging improvements

Stone log now includes command

The startup header printed in the Stone log now includes the command that was used to start the stone. For example:

```
| COMMAND: /gemstone/GemStone64Bit2.4.0/sys/stoned gs64stone
```

Backup start time now in stone log

The time a backup is started is now recorded in the stone log, in addition to the time that backup completes.

addAllToStoneLog now includes session information

When a message is printed to the stone log from a gem session using System class >> addAllToStoneLog:, the output header now includes the user name, session id, session serial number, and the PID of the gem.

Image upgrade now reported in Stone log

The image upgrade process now reports the upgrade in the Stone log.

New Reflection API

To allow access to information without sending a message to the object itself, a new kernel class has been added. The following class methods are available for the Reflection class:

`oopOf:` *anObject*
Answers the oop of *anObject*, consistent with `Object>>asOop`.

`classOf:` *anObject*
Answers the class of *anObject*, consistent with `GciFetchClass`.

`implementationOf:` *anObject*
Answers #pointers, #bytes, #NSC, or #special, consistent with `GciFetchObjImpl`.

`sizeOf:` *anObject*
Answers 0 if `implementationOf:` answers #special. Otherwise, answers the number of pointers (for #pointers or #NSC) or bytes (#bytes) that *anObject* currently holds. For #pointers objects (small and large), this includes both named instance variables and the object's varying size. Consistent with `GciFetchSize`.

`namedSizeOf:` *anObject*
Answers 0 if #implementationOf: answers #special. Otherwise, answers the number of named instance variables of *anObject*. Consistent with `GciFetchSize`.

`setSizeOf:` *anObject* `to:` *newSize*
Sets the indexable size of *anObject* so that it will subsequently answer *newSize* when sent sizeOf:. Consistent with `GciSetVaryingSize`. Not valid for #NSC or #special objects.

`fetchFrom:` *anObject* `at:` *anIndex*
Answers the object or byte found at the given index in *anObject*. Legal values for *anIndex* are 1 to sizeOf: *anObject*. Not valid for 0-size objects. Consistent with `GciFetchOop` and `GciFetchByte`.

`storeTo:` *anObject* `at:` *anIndex* `put:` *aValue*
Stores *aValue*, which may be any object for #pointers objects, but which must be 0-255 for #byte objects, at the given index in *anObject*. Consistent with `GciStoreOop` and `GciStoreByte`. Legal indices are 1 to sizeOf: *anObject*.

`to:` *anNsc* `addObject:` *anObject*
Add *anObject* to the NSC. Consistent with `GciAddOopToNsc`.

`from:` *anNsc* `removeObject:` *anObject*
Removes an existing object *anObject* from the NSC. Consistent with `GciRemoveOopFromNsc`. Answer true if *anObject* was in the NSC and has been removed, false if it was not present.

`fetchTagOf:` *anObject* `tagNum:` *tagNumber*
Answer the object referred to by the given tag number of *anObject*. *tagNumber* must be 1 or 2. Similar to primitive 87.

`storeTagTo:` *anObject* `tagNum:` *tagNumber* `newValue:` *aValue*
Store the given value into the specified tag number of *anObject*. *tagNumber* must be 1 or 2. Similar to primitive 280.

String byteAt: and byteAt:put: removed

Formerly, instances of String and MultiByteString (including DoubleByteString and QuadByteString) understood the methods `byteAt:` and `byteAt:put:`. However, these messages have potential issues with MultiByteStrings between big-endian and little-endian platforms. It is intended that Strings and MultiByteStrings be accessed via Character rather than by byte (partial Character). For byte-level manipulation, ByteArrays should be used.

To replace `byteAt:` and `byteAt:put:`, the methods `codePointAt:` and `codePointAt:put:` have been added. Both String and MultiByteStrings understand these methods. For MultiByteStrings, these methods differ from the **`byteAt:`** equivalent in that they return the value of the Character at that index in the String, rather than one byte of a Character that potentially may require 2 or more bytes to represent.

In addition, instance of MultiByteString no longer understand `withBytes:`. This method is still available for instances of String.

If your application depends on byte level access to MultiByteStrings, appropriate methods can be defined in the image. Contact GemStone Technical Support if you need assistance with this.

Note that JapaneseString and its subclasses now implement `byteAt:` and `byteAt:put:`. These classes are not within the MultiByteString hierarchy.

GciInterface methods _objectForOop:/_oopForObject:

The methods `GciInterface class >> _objectForOop:` and `GciInterface class >> _oopForObject:` now use the equivalent Object methods `Object class >> _objectForOop:` and `Object >> asOop`.

DateAndTime now and TimeZone

The result of `DateAndTime now` bases the time of the result on the current TimeZone for the gem, rather than as previously, the default TimeZone for the Stone. The ANSI specification permits either.

DateAndTime is the ANSI-specified class analogous to DateTime.

Additional key in gem and stoneVersionReport

The dictionary returned by `System >> gemVersionReport` and `stoneVersionReport` now includes the #gsBuildSerialNumber key. This is primarily for internal use and testing.

Change in behavior for loadGcCandidatesFromFile:

The method `Repository>>loadGcCandidatesFromFile:` *aString* `intoHiddenSet:` *hiddenSetId* previously did not validate the objects in the file. Now, only valid object which exist and are not in the dead set are loaded. With this change, this method can be used to determine how many objects in an FDC file would pass MGC validation.

This method now returns an Array of 3 SmallIntegers:
1 - Total number of objects in the file.
2 - Number of valid objects loaded into the hidden set.
3 - Number of invalid objects not loaded.

Improvements to Exception Handling

There have been changes and refactoring in the Exception handling framework. These are primarily internal changes and changes to fix bugs, and should not affect your application behavior provided you are using the interface documented in the Programmer's Guide for GemStone/S 64 Bit.

The category arguments for Exception handling methods, and for the topaz commands EXPECTERROR and EXPECTBUG, are no longer strictly restricted to SymbolDictionary. This is primarily an internal change and should not affect customer code. As a result, the SymbolDictionary ANSIException, which was added in version 2.3, is no longer needed. This SymbolDictionary is not present in new repositories or in repositories upgraded from versions earlier than 2.3 to version 2.4. In repositories that originated in or have been previously upgraded to 2.3.x, ANSIException will continue to exist, but it is no longer used.

The refactoring, and removal of some layers of indirection, resulted in a number of methods being removed. The removed methods were intended for internal use, although they were not marked private, and should not impact customer code.

Class and Method Changes

The class IgnoredException has been added, to allow handler search to skip active exceptions.

The following methods have been added in GemStone/S 64 Bit v2.4:

```
Error Class >>handles:
Exception>>category:
Exception Class >>errorCategoryToCatch
Exception Class >>errorNumberToCatch
ExceptionA Class >>handlesCategory:number:
ExceptionA Class >>errorCategoryToCatch
ExceptionA>>signal:with:
ExceptionA>>textForError:args:
ExceptionA Class >>signal:with:
ExceptionA Class >>signalWith:
ExceptionHandler>>caughtEx:number:cat:args:handler:
ExceptionHandler>>caughtExceptionWithAction:
ExceptionHandler>>doTryBlock:
ExceptionHandler>>pass
ExceptionHandler>>resignalAs:
ExceptionHandler>>retryUsing:
ExceptionHandler>>try:on:do:
ExceptionSet>>errorCategoryToCatch
ExceptionSet>>errorNumberToCatch
ExceptionSet>>handlesCategory:number:
SymbolDictionary>>errorCategoryToCatch
SymbolDictionary>>errorNumberToCatch
ZeroDivide Class >>errorCategoryToCatch
MessageNotUnderstood Class >>errorCategoryToCatch
```

The following methods have been removed since version 2.3.1.6:

```
Exception Class >>try:on:do:
ExceptionA>>action
ExceptionA>>action:
ExceptionA>>doResignal
ExceptionA>>doResignalAs
ExceptionA>>doRetry
ExceptionA>>firstGSException
ExceptionA>>result
ExceptionA>>result:
ExceptionA>>shouldReturnFromHandlerBlock
ExceptionA>>valueHandler:
ExceptionA Class >>exceptionCategory
ExceptionA Class >>initialize
ExceptionA Class >>staticCatch:category:number:arguments:
ExceptionA Class >>staticExceptionBlock
ExceptionA Class >>try:on:do:
ExceptionHandler>>caughtEx:number:cat:args:
ExceptionHandler>>caughtException
ExceptionHandler>>exceptionSelector:
```

```
ExceptionHandler>>handlerBlock
ExceptionHandler>>handlerBlock:
ExceptionHandler>>outerFrom:
ExceptionHandler>>shouldWeHandleThis
ExceptionHandler>>signalEarlyReturn
ExceptionHandler>>signaledException
ExceptionHandler>>signaledException:
ExceptionHandler>>try:on:in:do:
ExceptionHandler>>tryBlock
ExceptionHandler>>tryBlock:
ExceptionHandler>>valueHandlerBlock
ExceptionSet>>try:on:do:
SymbolDictionary>>try:on:do:
```

SigAbort is now resumeable

SigAbort, and certain other errors, are now resumeable. (#40152)

Changes in Utilities

startstone -N now ignores existing transaction log

Previously, when restarting the stone using the -N option to not recover using the transaction logs, it was a requirement that the existing transaction logs be removed first. Now, startstone with the -N option will ignore existing transaction logs and start up without recovery.

startcachewarmer

This utility now includes the -T *int* option, which sets the TemporaryObjectCache size for the cache warmer gems. By default, if this option is not specified, the TOC is set to 5MB.

Configuration Parameter Changes

Modified configuration Parameters

STN_MAX_AIO_RATE

The new maximum value for this parameter is 1000000.

Added configuration parameter

GEM_TEMPOBJ_POMGEN_SCAVENGE_INTERVAL

Interval in seconds in which the oldest pom gen subspace will be discarded. Lower values may reduce gem memory usage but may also cause objects to be re-read. Larger values may result in higher gem memory usage and may reduce disk I/O. Setting this value to zero disables scheduled pom gen scavenges. In this case, pom gen will only be scavenged when all subspaces become full.

In versions prior to 2.4, this value was set to 240 seconds.

Runtime equivalent: #GemTempObjPomgenScavengeInterval
Default: 1800
Min: 0
Max: 86400
Units: seconds

Topaz Changes

Added LOOKUP command

A new topaz command has been added, LOOKUP, similar to LIST.

LOOKUP method *aSelector*
LOOKUP classmethod *aSelector*

Using the current class as set by a previous SET CLASS, search for the method *aSelector* going up the class hierarchy. Print the class in which the method found, and list the source code.

EXPECTBUG, EXPECTERROR

The category argument may now be %SymbolDictionary, as well as the specific instance of SymbolDictionary, GemStoneError.

Added SET CACHENAME option

The ability to set the cache statistics name in topaz prior to login has been added. This avoids the session displaying two names in the final statistics, when setting the cache name after login.

This feature applies to linked sessions only.

Cache Statistics Changes

You must use the version of VSD that is distributed with GemStone/S 64 Bit v2.4 or later, in order to read the cache statistics generated by this version.

Range changes

Cache statistics for Gems, and GlobalSessionStats for the Stone, are now 64 bit, and can hold 64-bit Integer values.

The interface methods accept arguments in the full SmallInteger range, up to 2^{60} -1 or 1152921504606846975. Once a cache statistics is set, it can be incremented beyond the SmallInteger range, up to 2^{64} -1. Increments that should result in values greater than 64 bits do not error, but result in invalid (negative) values.

VSD has been updated to ensure 64-bit values are handled correctly.

The following methods now accept a larger range of argument values:

`System class >> _sessionCacheStatAt:` *anIndex* `put:` *aSmallInt*

Previously, this method required that the argument aSmallInt be a positive SmallInteger. Now, negative SmallIntegers are permitted.

`System class >> _updateGlobalSessionStat:` *index* `by:` *aSmallInt* `overwrite:` *aBoolean*

Previously, this method required that the value *aSmallInt* be in the range -2147483648 to 2147483647. Now, any SmallInteger is permitted.

Added methods

The following methods have been added:

`System class >> _sessionCacheStatAt:` *anIndex* `incrementBy:` *aSmallInt*
Increment the cache statistic at the given index by *aSmallInt*.

`System class >> _sessionCacheStatAt:` *anIndex* `decrementBy:` *aSmallInt*
Decrement the cache statistic at the given index by *aSmallInt*.

`System class >> myCacheStatistics`
Return the statistics for the current sesssion.

Programmatic access to sysTime and userTime

On systems that support host statistics, the array returned by `System >> cacheStatistics:` now also includes userTime and sysTime, in milliseconds.

Renamed cache Statistics

The SPC Monitor cache statistic **SlotsRecoveredCount** has been renamed to **CrashedSlotsInRecoveryCount**.

The Stone cache statistics **StnAioSuspendEAGAIN** has been renamed to **StnAioSuspendTimeoutCount**.

Added Cache Statistics

The following statistics have been added.

BackupHighWaterPage (Stone)
The first page ID which reclaim gems are allowed to reclaim during a full backup.

CacheSlotIndex (All)
Index of the slot in the shared page cache used by this process.

CleanSlotsInRecoveryCount (SPC Monitor)
The current number of slots being recovered which were owned by a process which shutdown cleanly.

CleanSlotsRecoveredCount (SPC Monitor)
The total number of slots the shared cache monitor has recovered because a client process shutdown cleanly.

CleanSlotThreadTimeRunningMs (SPC Monitor)
Approximate real time in milliseconds that the clean slot recovery thread in the shared page cache monitor has spent doing work.

CodeGenClearSendCachesCount (Gem)
Total number of times the gem has cleared all method send caches in code gen.

CrashedSlotsRecoveredCount (SPC Monitor)
The total number of slots the shared cache monitor has recovered because a client process shutdown abnormally.

CrashedSlotThreadTimeRunningMs (SPC Monitor)
Approximate real time in milliseconds that the crashed slot recovery thread in the shared page cache monitor has spent doing work.

GarbageCollectionState (Gem)
Indicates the state of a garbage collection task in the Admin GcGem or a user performing a garbage collection operation. Values are defined as follows:

0 - Garbage collection not active
1 - Initialization
2 - Mark-sweep
3 - Compute dead objects
4 - Remove not dead objects
5 - Finalize weak references
6 - Start record dead
7 - Write possible dead objects to tranlog
8 - Finish record dead
9 - Write not dead objects to tranlog
10 - Send not dead objects to stone

EpochLastDuration (Stone)
Duration of the last completed epoch garbage collection operation in seconds.

GcHighWaterPage (Stone)
The first page ID which reclaim gems are allowed to reclaim during garbage collection or other repository scan.

GcWsUnionSize (Stone)
Number of objects in the write set union used to finalize possible dead objects. All

objects in the WSU must be scanned to determine if any objects reference one or more possible dead objects.

GemTempObjCacheSizeKb (Gem)
The value of the session's GEM_TEMPOBJ_CACHE_SIZE_KB configuration parameter.

MainThreadTimeRunningMs (SPC Monitor)
Approximate real time in milliseconds that the main shared page cache monitor thread has spent doing work.

ObjectsReadInBytes (Gem)
Total size in bytes of committed objects copied into VM memory since start of session.

PrimitiveNumber (Gem)
The primitive number currently being executed by the session, or 0 if the session is not in a primitive. The session only sets this value for long-running primitives. A non-zero value also indicates the session is immune from termination due to the STN_GEM_TIMEOUT mechanism.

Note: the PrimitiveNumber stat is set to 9999 if the session has executed System class >> disableStoneGemTimeout

RecoverFreeFrameWaitTime (Stone)
The time in seconds spent by the main recovery thread waiting for free frames in the shared page cache. Recovery performance may be improved by increasing the size of the shared page cache.

RecoverNumBufs (Stone)
The total number of logEntryBuffers allocated in the heap.

RecoverNumBufsForSessions (Stone)
The number of logEntryBuffers used to hold records for sessions not yet committed.

RecoverNumBufsInFreeList (Stone)
The number of logEntryBuffers currently in the freeList.

RecoverNumBufsInWorkQueue (Stone)
The number of logEntryBuffers currently in the workQueue.

RecoverReadThreadWaitTime (Stone)
The time in seconds that the reader thread spent waiting for the main recovery thread to catch up processing the log buffers.

RecoverReclaimOopsWaitTime (Stone)
The time in seconds spent by the main recovery thread waiting for oops to be reclaimed. Recovery performance may be improved by adding more reclaim gems.

RejectedProcsCount (SPC Monitor)
The total number of processes which attempted to connect to the shared page cache but were rejected because the maximum number of processes had already attached.

SlotsFreeCount (SPC Monitor)
The number of free process slots currently available in the cache.

SlotsTotalCount (SPC Monitor)
The maximum number of processes that can concurrently attach to the shared page cache.

StatsThreadTimeRunningMs (SPC Monitor)
Approximate real time in milliseconds that the statistics thread in the shared page cache monitor has spent doing work.

StnAioCompleted1Suspend (Stone)
Number of AIO requests completed after one call to aio_suspend().

StnAioCompletedNoSleep (Stone)
Number of AIO requests that were completed without sleeping. The AIO thread in stone will sleep for a short time if aio_suspend() returns prematurely several times.

StnAioCompletedNoSuspend (Stone)
Number of AIO requested completed without calling aio_suspend().

StnAioLastSuspendCount (Stone)
Number of times that aio_suspend() was called to complete the last AIO request.

StnAioMainTimeInAioWrite (Stone)
Total real time in milliseconds that the main thread spent executing the aio_write() call. aio_write() is used to initiate asynchronous writes to the tranlog and should not block.

StnAioTotalSleepCount (Stone)
Total number of times the AIO wait thread in stone went to sleep while waiting for an AIO to complete.

StnAioWaitLastTime (Stone)
Time in microseconds that it took the AIO thread to complete the last AIO request.

StnAioWaitTotalTime (Stone)
Approximate total real time in milliseconds that the AIO wait thread spent waiting for asynchronous I/O requests to complete.

StnCrBacklogThreshold (Stone)
Current setting of the STN_CR_BACKLOG_THRESHOLD stone configuration parameter.

TimeSleepingMs (Pgsvr)
Total real time the process has spent sleeping, in milliseconds.

TotalProcsInCacheCount (SPC Monitor)
The total number of processes currently connected to the cache, including crashed processes that have not yet had their cache slot recovered.

Bugs Fixed

The following bugs in GemStone/S 64 Bit 2.3.1.6 have been fixed in GemStone/S 64 Bit 2.4.

Performance problems due to AIO behavior

Under certain conditions of heavy load, system performance could become slow due to AIO delays. These problems are related to the behavior of aio_suspend on Solaris. As a result, the AIO wait behavior has been redesigned. See “AIO wait behavior” on page 2. (#39738, #40124)

After AIO operation cancelled, repeated error in stone log

After recovering from AIO issues, the flag indicating a AIO operations was being cancelled could get stuck. This resulted in many repeated error messages in the stone log, which did not reflect any further actual errors. (#39738)

Garbage Collection related issues

Dead object reclaim may pause for 5 minutes

If there is a large amount of shadow reclaim to do when objects are promoted to dead, the Reclaim GcGems may continue to reclaim shadow objects, and not check for new dead objects to reclaim. After 5 minutes, the Stone forces the Reclaim GcGems to update their dead set, triggering dead reclaim. (#39901)

MFC could have ignored sigAborts during initial stage

SigAbort signals that arrived for the MFC gem during a (normally brief) initial stage of MFC could be ignored. Followup sigAborts are not sent, which could have resulted in a commit record backlog. #40122)

MFC ran in transaction on Linux

Linux only

On Linux only, a gem running a markForCollection (MFC) did not adjust its transaction mode. If the session was in transaction prior to starting MFC, the MFC would run in transaction, not aborting and potentially causing commit record backlogs and consuming excessive free space. (#40178)

MGC may have failed with a commit error

A markGcCandidates (MGC) operation could fail with a commit error, although the commit did not actually fail and the MGC should have completed. (#39934)

GBS errors on large traversals

GBS 7.2 and above only

When GBS performed a GciStoreTravDoTravRefs call with more than 2034 dirty server objects, internal code reading from a bitmap only read the first 2034 OOPs. This caused GBS to temporarily lose synchronization with the server. This could result in GBS errors and/or incorrect object state. (#39853)

Out of memory errors

There were several circumstances under which it was possible to get out of memory errors.

If a large amount of code was compiled, under some circumstances the code-gem space garbage collection did not keep up, resulting in an out of memory error when this space became full (#40042)

The additional memory required during commit in certain cases (most notably, indexing operations) could cause an out of memory error. (#40130)

Unnecessary memory warnings when handling almostOutOfMemory

When running too close to session temporary memory limits, by default stacks containing details of memory use are printed to the session's log. If the almostOutOfMemory signal is enabled and handled, these stacks are unnecessary and alarming, and may obscure actual issues. Now, if the almostOutOfMemory signal is enabled, these stacks are not printed to the log.(#40103)

Indexing related bugs

SelectBlock queries may encounter message not understood error

Certain selectBlock multi-term queries which include a range predicate may the incorrect method names, resulting in a does not understand error. (#39980)

Index creation could be killed by timeout

A gem doing index creation that took a long time could have been timed out and terminated by a STN_GEM_TIMEOUT setting. (#39941)

Now, the IndexManager uses the new feature to disable STN_GEM_TIMEOUT and will not be killed by this timeout. See "Ability to disable STN_GEM_TIMOUT" on page 9 for details on this new feature.

Index audit errors if elements invariant

If the elements in an indexed collection are invariant, `UnorderedCollection >> auditIndexes` may return spurious errors. (#40076)

Index audit did not catch corruption in Btree cached values

Btrees cache a portion of the value within their structures, to allow faster searches since the actual value object does not need to be fetched. The index audit code used the actual value, not the cached value, for auditing, and did not report problems where the actual value was correct but the cached value in the btree was incorrect. (#39943)

Gem could have core dumped on page server death

If a page server was killed, a linked gem could have core dumped, rather than died normally. (#39723)

Signal handling issues

Risk of Stone hang or crash on Gem SIGTERM

On a heavily loaded system, if a Gem on the same host as the Stone gets a SIGTERM while waiting for a shared memory (SMC) response from the Stone, there is a possible race condition that can hang or bring down the Stone. This is because the same semaphore is used to signal Gems waiting for SMC communications, and waiting for spin locks.

The SIGTERM interrupts the Gem's wait for the SMC response from the Stone, and the Gem continues handling the SIGTERM, which can involve waiting on spin locks to get page frames, etc. as part of shutdown. However, when the Stone finally completes the SMC response and signals the Gem, the Gem assumes this is the signal that the spin lock it was waiting for is now is available, resulting in spin lock problems that may hang or crash the Stone, depending on the specific spin lock. (#40096)

Incorrect signal handler for SIGINT

The signal handler installed for SIGINT specified an invalid number of arguments. (#39681)

Signal handler chaining could pass incorrect number of arguments

When signal handlers are chained, either one or three arguments should be passed, specified in SA_SIGINFO. GemStone signal handlers did not check, and always passed three arguments, which could be incorrect for customer installed signal handlers. (#39672)

Login and password related issues

Incorrect reason for disabled account

After logins are disabled, the reason is recorded on the user’s account, and further attempts to log in will fail. If the user then attempted to log in, after the configured number of failed login attempts, the original reason for the account disable was overwritten by LoginsWithInvalidPassword. (#39521)

UserProfile >> isDisabled may not have returned correct information

A UserProfile using password age or login time expiration limits will be suspended the next time that user attempts to login after the time limit is exceeded. But until that login attempt, the `UserProfile >> isDisabled` method continued to return false for that UserProfile. (#39920)

Risk of SEGV in cache warmer gems

AIX only

The last cache warmer gem to exit may have crashed due to an internal computed integer overflow. (#39695)

Space(s) in cache name corrupted statmonitor data file

Previously, assigning a cache name that included spaces corrupted the statmonitor file. Now, Statmonitor replaces space characters with underscore characters. (#39725)

GsFile Bugs

GsFile >> atEnd returned true for compressed files

The method `GsFile >> atEnd` always returned true for compressed files, regardless of position. (#39663)

Note that `atEnd` is not supported for files opened for write. If the compressed GsFile was opened for write, `atEnd` may continue to always return true.

GsFile >> contents returned nil for zero sized files

If a file was empty, `GsFile >> contents` returned nil, rather than an empty string. (#40001)

Integer >> to: did not return Interval

The method `Integer >> to:` returned an Array, while `Integer >> to:by:`, and non-Integer number implementations of `to:` and `to:by:`, return Intervals, per the ANSI spec. `Integer >> to:` now also returns an Interval. (#40104)

findReferencePathToObject: can produce corrupt object error

When a reference path found while executing `findReferencePathToObject:` included a large collections, it could have returned a spurious corruption object error. (#39727)

Incorrect restore point in timeToRestoreTo:

When the DateTime specified as an argument to `Repository >> timeToRestoreTo:` has a different offset from GMT as the current time, the calculation of the seconds offset, used internally to perform the restore, was incorrect by the amount of offset. This occurs for example when one time is during Daylight Savings Time and the other is not. (#39636)

Sesssions tied up after stone restart with extent resize

If the stone is started up with larger extents specified in the configuration file than the sizes on shutdown, and DBF_PRE_GROW is true, the extent files are resized to the new size. This resize inadvertently tied up session slots, resulting in fewer than the configured number of sessions able to log in until the subsequent stone restart. (#40149)

addAllToStoneLog: errors for very large argument

If the argument to `System >> addAllToStoneLog:` was larger than 16270, this method returned an error. (#39802)

Spurious conflicts for commits with commitCoordinator

GemStone has the ability for a pseudo two-phase commit using a commitCoordinator. This mechanism is used by GemConnect. If a commitCoordinator is set up, with several other conditions, it is possible for commits involving indexing structures to return commitConflicts errors without any actual commit conflicts.   (#40094)

While this scenario has been fixed, GemStone does not yet support a true two-phase commit.

Object does not exist errors on _getAndClearGciDirtySet:into:

When enumerating the dirty or tracked objects set using `System class >> _getAndClearGciDirtySet:into:`, chunks of 2034 objects are returned. If there are more than 2034 objects and objects to be garbage collected between iterations, it was possible to get object does not exist errors. (#40276)

To address this problem, objects in the dirty or tracked objects sets which no longer exist are silently ignored. Objects in the dirty set which still exist but are no longer in the export set are also silently ignored, since an object may be garbage collected and reused at any time after its removal from the export set.

Collection issues

deepCopy sent do:, potential side effects

The `deepCopy` implementation for Collections sent `do:` to iterate the contents of the collection. This method could be overridden by subclasses and result in undesired side effects. (#39958)

SortedCollection had incorrect Identity-based behavior

The ANSI standard calls for SortedCollections to be equality-based. Some methods invoked on SortedCollection instances, such as `indexOf:` and `remove:`, were incorrectly identity based.(#39796)

SortedCollection reverse errors

Sending `reverse` to an instance of SortedCollection generated an error, since reverse would violate the order of the sortBlock. Now, *aSortedCollection* `reverse` returns an OrderedCollection containing the elements of the SortedCollection in the reversed order. (#40040)

String >> addAll: failed for collections of Characters

Per ANSI specification, `String >> addAll:` should accept Collections of Characters as an argument. Previous, only instances of CharacterCollection or its subclasses were accepted, or Characters. (#39794)

FastIdentityKeyValueDictionary privilege checking

Due to inherited methods, calls to instances of this class were doing privilege checks, slowing performance. (#40105)

CollisionBuckets did not understand do:

Iteration over instances of subclasses of AbstractCollisionBucket would result in an error. (#39992)

IdentitySet asSet failed

Conversion of an IdentitySet to a Set using `asSet` was inappropriately disallowed. (#39587)

upTo:do: incorrect end-of-stream error

The method `PositionableStream >> upTo:do:` could return an end of stream error when argument match was at the end of the receiver Stream. (#39687)

Conversion issues

Incorrect SystemUser default segment causes conversion problems

If the default segment for SystemUser is set to a segment other than SystemSegment, it may cause serious problems with conversion or upgrade, as the fileins place kernel code in this segment. (#39656)

DataCuratorGroup missing on converted repositories

Repositories converted from GemStone/S 64 Bit v1.x or 2.0.x, or from GemStone/S, did not have the DataCuratorGroup in the DataCurator’s groups. (#39834)

Minor differences in classes between upgraded and new repositories

Repositories that were upgraded from earlier versions showed several minor differences: ClampSpecification had an incorrect constraint, ProfMonitor retained obsolete class variables, and the class comment for Repository was incorrect. (#39858, 39834)

Errors after upgrade with customized Extended Character Sets

Version 2.3 included updates to the storage format used for installed customized Extended Character Sets to accommodate QuadByteStrings. Upgrading from a version 2.2.x or GemStone/S 6.2.x, 6.3.x, or 6.5.x repository with a customized Extended Character Set installed resulted in errors in the upgraded/converted repository. (#40242)

DateTimes created with milliseconds >= 2^{31} lose precision

If the millisecond field in a DateTime is set to a value equal to or greater than 2^{31}, the resulting DateTime loses the millisecond component. (#40273)

Transactionless sessions result in shortened LostOT timeout

If all gems are in transactionless mode, the lostOT timeout interval is reduced. This was intended to keep pooled gems more current, but it impacted customer uses of transactionless mode. Now, the system makes an additional check that all sessions are logged in as Nameless before reducing the timeout. (#39678)

STN_COMMIT_QUEUE_THRESHOLD configured value not used

The configurative parameter STN_COMMIT_QUEUE_THRESHOLD is overwritten internally and the value not used, although a configured setting may be reported in the stone log and returned by the runtime query. (#40255)

Recluster after abort resulted in error

A second cluster operation following an abort would result in an error “No method was found for the selector #'_generality' when sent to nil”. (#39753)

GsSocket connectTo: errors

If `GsSocket >> connectTo:` encounters problems where it cannot connect, it should return false, but was incorrectly raising an error. (#40134)

pageaudit -f could encounter errors but report success

The pageaudit -f option allows the pageaudit to continue running after discovering an error. If there were errors, and pageaudit -f continued to run further audit steps, the final result could have been incorrectly reported as success. (#40160).

Invalid compiler warnings for inefficient temporary variable

The compiler may issue a warning about an inefficient temporary variable - that is, a variable in an outer scope that could be moved to an inner scope - in cases when the variable could not be moved. (#39774)

tranlog analysis scripts could use incorrect temporary directory

UNIX/Linux systems may use one of several locations for temporary files. To avoid errors if a particular temporary file directory is not available in a particular installation, the script now looks for the following directories, in order, and uses the first one found for temporary files: `/usr/tmp`, `/var/tmp`, and `/tmp` (#40186)

Unnecessary warnings with tranlogs on raw partitions

When tranlogs were on raw partitions whose path did not start with `/dev`, invalid warnings were printed in the stone log on stone startup. (#39760)

Script errors accessing uname on the Macintosh

Macintosh only

GemStone scripts were expecting the uname utility to be in the `/bin` directory. On the Macintosh, the location of the uname utility may be in `/usr/bin`, rather than `/bin`. The scripts now check both locations. (#40199)

TimeZone class >> fromOS failed on the Macintosh

Macintosh only

The method `TimeZone class >> fromOS` did not handle the Macintosh platform correctly. (#40213)

sourceToFirstComment errors for overly empty methods

Methods whose source string includes no method body, including no newline following the selector, get an out of range error when sent `sourceToFirstComment` (#39267)

Default for GS_DEBUG_VMGC_MKSW_MEMORY_USED_SOFT_BREAK

In the gemnetdebug script, the environment variable GS_DEBUG_VMGC_MKSW_MEMORY_USED_SOFT_BREAK was enabled by default, which inconveniently introduced soft breaks in unrelated debugging scenarios. Now, if you are specifically debugging out of memory issues and need this environment variable, edit the gemnetdebug script to remove the comments and set this variable. (#40282)