GemStone/S 64 Bit™
Release Notes

Version 3.4

October 2017

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

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

GemTalk, **GemStone**, **GemBuilder**, **GemConnect**, and the GemTalk logo are 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.

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

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

Microsoft, **Windows**, and **Windows Server** 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.

Ubuntu is a registered trademark of Canonical Ltd., Inc., in the U.S. and other countries.

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

AIX, **POWER6**, **POWER7**, and **POWER8** and **VisualAge** are trademarks or registered trademarks of International Business Machines Corporation.

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

CINCOM, **Cincom Smalltalk**, and **VisualWorks** are trademarks or registered trademarks of Cincom Systems, Inc.

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 LLC
15220 NW Greenbrier Parkway
Suite 240
Beaverton, OR 97006

Preface

About This Documentation

These release notes describe changes in the GemStone/S 64 Bit™ version 3.4 release. Read these release notes carefully before you begin installation, conversion testing, or development with this release.

For information on installing or upgrading to this version of GemStone/S 64 Bit, please refer to the *GemStone/S 64 Bit Installation Guide* for version 3.4.

For questions or to submit feedback on this manual, join the documentation mailing list: http://lists.gemtalksystems.com/mailman/listinfo/documentation.

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 LLC, previously GemStone Systems, Inc. and a division of VMware, Inc.

Technical Support

Support Website

gemtalksystems.com

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

- **Documentation** for the current and for previous released versions of all GemTalk products, in PDF form.
- **Product download** for the current and selected recent versions of GemTalk software.

- **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.

We recommend checking this site on a regular basis for the latest updates.

Help Requests

GemTalk Technical Support is limited to customers with current support contracts. Requests for technical assistance may be submitted online (including by email), or by telephone. We recommend you use telephone contact only for urgent requests that require immediate evaluation, such as a production system down. The support website is the preferred way to contact Technical Support.

Website: techsupport.gemtalksystems.com

Email: techsupport@gemtalksystems.com

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

Please include the following, in addition to a description of the issue:

- The versions of GemStone/S 64 Bit and of all related GemTalk products, and of any other related products, such as client Smalltalk products, and the operating system and version you are using.
- Exact error message received, if any, including log files and statmonitor data 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.

Table of Contents

Chapter 1. GemStone/S 64 Bit 3.4 Release Notes

Chapter 2. Changes in Administration

Chapter 3. GemStone Smalltalk changes and new features

Chapter 4. Changes in the GCI Interfaces

Chapter 5. Bug Fixes

Chapter

1 GemStone/S 64 Bit 3.4 Release Notes

Overview

GemStone/S 64 Bit™ 3.4 is a new version of the GemStone/S 64 Bit object server. This release includes many new features, provides improved performance, and fixes a number of bugs. We recommend everyone using or planning to use GemStone/S 64 Bit upgrade to this new version.

These release notes describe changes between the previous version of GemStone/S 64 Bit, version 3.3.6, and version 3.4. If you are upgrading from a version prior to 3.3.6, review the release notes for each intermediate release to see the full set of changes.

For details about installing GemStone/S 64 Bit 3.4 or upgrading from earlier versions of GemStone/S 64 Bit, see the *GemStone/S 64 Bit Installation Guide* for v3.4 for your platform.

Upgrade

Upgrade is supported from GemStone/S 64 Bit v2.4.x, 3.1.x, 3.2.x. and 3.3.x. For details on the conversion and upgrade, see the *GemStone/S 64 Bit Installation Guide* for v3.4.

New keyfiles required

New keyfiles are required for version 3.4. To obtain a new keyfile for GemStone/S v3.4, write to keyfiles@gemtalksystems.com. In your request, include 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.

Supported Platforms

Platforms for Version 3.4

GemStone/S 64 Bit version 3.4 is supported on the following platforms:

- Red Hat Enterprise Linux Server 6.7 and 7.1, Ubuntu 14.04 and 16.04, and SUSE Linux Enterprise 12, on x86
- Solaris 10 and 11.3 on x86
- AIX 6.1 on POWER6 and POWER7 and AIX 7.1 on POWER8
- OS X 10.11.2 (El Capitan) with Darwin 15.2.0 kernel, and OS X 10.12.6 (Sierra), with Darwin 16.7.0 kernel, on x86 (Mac is supported for development only)

Note that Solaris on SPARC is no longer a supported platform, although distributions are available for development or debugging.

For more information and detailed requirements for each supported platforms, please refer to the GemStone/S 64 Bit v3.4 Installation Guide for that platform.

GemBuilder for Smalltalk (GBS) Versions

GemStone/S 64 Bit version 3.4 requires GBS version 8.3 or later for VisualWorks Smalltalk, or version 5.4.4 or later for VA Smalltalk.

The following versions of GBS are supported with GemStone/S 64 Bit version 3.4:

GBS version 8.3

VisualWorks 8.2.1 32-bit and 64-bit	VisualWorks 7.10.1 32-bit	VisualWorks 7.10.1 64-bit
▸ Windows 10, Windows 8, Windows 2008 R2 and Windows 7 ▸ RedHat ES 6.7 and 7.1, and Ubuntu 14.04 and 16.04	▸ Windows 10, Windows 8, Windows 2008 R2 and Windows 7 ▸ RedHat ES 6.7 and 7.1, and Ubuntu 14.04 and 16.04	▸ Windows 10 ▸ RedHat ES 7.1

GBS version 5.4.4

VA Smalltalk 8.6.3
▸ Windows 10 ▸ Windows 8, Professional or above ▸ Windows 2008 R2 ▸ Windows 7, Professional or above

For more details on supported GBS and client Smalltalk platforms and requirements, see the *GemBuilder for Smalltalk Installation Guide* for that version of GBS.

VSD Version

The GemStone/S 64 Bit v3.4 distribution includes VSD version 5.3.1; The previous version of GemStone/S 64 Bit, v3.3.6, included VSD v5.3. Changes between these versions are minor, and include:

- updated zlib libraries.
- added definitions for new cache statistics.
- a bug fix in reading empty timestamps in statmonitor data records.

See the Release Notes for VSD v5.3.1, or the VSD home page.

Other Products

The current versions of GemBuilder for Java (GBJ) and GemConnect can be used with GemStone/S 64 Bit v3.4.

Documentation Changes

All documentation has been revised for this release, with modifications to incorporate new and changed features, as well as corrections and improvements.

In addition to the maintenance changes, the following improvements have been made:

- The *GemBuilder for C* manual is now is now available in .html as well as .pdf, and has been extensively updated for clarity and organization. The first chapter has been split into an introductory and a programming chapter. Compile and link information is by platform rather than by operation.
- Chapters 1-4 of the *System Administration Guide* have been extensively reorganized, and material moved to two added chapters: an introductory chapter and a separate chapter for the NetLDI and interprocess connectivity. Some material has been moved to the expanded NRS appendix.
- The *Programming Guide* chapters on Collections, Strings, and Indexing have been extensively revised for clarity and organization, as well as including new material. More information is provided on behavior for repositories in Unicode Comparison Mode.

1.1 Library, version and distribution changes

Updated library versions

- The version of OpenSSL has been updated to v1.1.0f.
- The version of Zlib has been updated to 1.2.11.
- ICU version 58.2 is included (in addition to the older ICU versions), and used for new repositories.
- The version of OpenLDAP has been updated to 2.4.45
- The version of LZ4 has been updated to 1.7.5

Library distribution change

Previously, the GemStone distribution included separate shared libraries for OpenSSL and LDAP, and (in 3.4 alpha versions) Kerberos. These libraries are now combined into a single file, libfloss (floss stands for Free/Libre Open Source Software).

This does not apply to Windows clients; there are no changes in distribution of Windows client libraries, other than version numbers.

The ICU libraries are not included in the libfloss library, since GemStone may load different versions of the ICU library.

The ICU libraries are now combined into a single file, rather than in three separate files as in previous releases. This only applies for the latest version of ICU, v58.2. The v3.4 distribution includes ICU v51.2 (for 3.2.x) and 54.1 (for 3.3.x), which continue to be distributed as three separate files.

Updated ICU shared library loading

GemStone/S 64 Bit v3.4 includes libraries for ICU versions 58.2, 54.1 and 51.2. Control over the ICU version that is used is managed similarly to the simplified process introduced in v3.3.5.

On AIX, is it no longer required to perform the manuals steps to ensure the correct libraries are loaded; ICU library loading is handled on AIX as it is on other platforms.

No environment variable or configuration parameter is required. The global variable (Globals at: #IcuLibraryVersion) controls the version of the shared library that is loaded. During upgrade, no special processing is needed; the code determines your previous version and sets (Globals at: #IcuLibraryVersion) appropriately.

However, there are a large number of possible upgrade paths. If you have created persistent structures depending on collation using Unicode strings, or any strings if the application in Unicode Comparison Mode, in v3.1.x or 3.0, you may need to rebuild/resort these structures. Contact GemTalk Technical Support if you think this case may apply.

The following ICU versions will be used:

- (Globals at: #IcuLibraryVersion), if set, will override the version set during upgrade.
- New applications (originating in v3.4) will use ICU v58.2, as will applications upgraded from 2.x or 3.0.x directly to 3.4.

- (Globals at: #IcuLibraryVersion) is automatically set during upgrade to v3.3.1 or later. Applications upgraded from 3.3.x to 3.4 will use ICU v54.1 or 51.2, depending on the previous upgrade.
- Applications upgraded directly from v3.2.x to 3.4 will use ICU v51.2.
- Applications upgraded directly from v3.1.x will use ICU v58.2, and require rebuild/resort of ICU-dependent structures.

During the startstone step of upgrade, information regarding the handling of the IcuLibraryVersion, and related processing, is written to the SymbolGem log (*stoneName_PID*`symbolgem.log`). This upgrade SymbolGem log is not deleted as is normally the default for SymbolGem logs.

Updated ZoneInfo TimeZone database

GemStone includes the zoneinfo TimeZone database. The has been update to version 2017d.

Updated SSL example certificates

GemStone distribution includes OpenSSL example certificates under:

```
$GEMSTONE/examples/openssl
```

These are used in examples for secure backup and GsSecureSocket example methods.

All the example certificates have been replaced. The directory structure includes an expanded directory structure as produced by openSSL.

There are now multiple client server example certificates, private keys, and password files. These certificates have been updated with improved security:

- 4096 bit RSA keys; previously they were 1024 bit.
- sha256 hashing; previously they were sha1.
- Add strong 32 character random passwords for private keys; previously they were short, predictable passwords.
- keys without passphrases are no longer distributed.

Configuration file

Several OpenSSL configuration files have been condensed down to one:

```
$GEMSTONE/examples/openssl/config/openssl.cnf
```

Scripts added

OpenSSL example keyfiles and scripts are in the distribution under `#$GEMSTONE/examples/openssl`.

The previous script to create examples, `make_example_certs.sh`, is no longer distributed. Now, script is provided to create a new CA (certificate authority):

```
create_ca.sh
```

And to create new certificate, private key and strong random password:

```
create_new_cert.sh
```

1.2 Keyfiles

New keyfiles are required for version 3.4, as mentioned on page 15. There are further changes that affect keyfiles.

Web/Community Edition license distribution on non-GLASS platforms

The GLASS/Seaside open source project and the Web Edition/Community Edition licenses are only supported for Linux and Macintosh. The free license distributed under the $GEMSTONE/seaside directory is no longer included with the AIX and Solaris distributions.

Option to update the keyfile without stopping the stone

Applications that do not have scheduled downtime may not easily be able to install a new keyfile with, for example, an updated Sunset date.

Methods have been added to allow a new keyfile to be applied without the need to shutdown and restart the stone.

`System class >> applyKeyfileNamed:` *keyfilenameOrNil*
Causes the stone to read the keyfile *keyfilenameOrNil*. If nil, the stone will use the same file name used when the stone was started.

The new keyfile permissions apply only as long as the stone is running. To permanently use the new keyfile, modify the KEYFILE option in the stone configuration file.

Returns true upon success. In this case, the keyfile was read and one or more changes were made to the system. Returns false if the keyfile was successfully read but contained no meaningful changes from the current configuration.

Raises an exception if the keyfile could not be read, has expired, or if applying it would modify parameters which may not be changed at runtime.

`System class >> getKeyfileAttributes`
Returns a SymbolDictionary which contains attributes of the current keyfile. See the image method for specific keys.

1.3 Optimizations

Primitives optimized

A number of primitives, including often-used ones like `at:`, have been optimized for better performance.

TLS/SSL now used over remote connection from GCI client to NetLDI

TLS/SSL is now enabled for the GCI client to NetLDI connection if the GCI is on a different machine than the NetLDI.

Note that this effects the handling of client library version mismatch errors; on a version mismatch between 3.4 and an older version, the error reported will be an SSL error or similar handshake error, rather than a version mismatch.

Transport Layer Security (TLS) is the successor to Secure Sockets Layer (SSL). GemStone continues to use the term "SSL" in most cases; SSL is commonly used to refer to both, and is more widely known.

More efficient transaction logging

A number of changes have been made to reduce the volume of tranlog data related to updates to collections, and new primitives have been added to support this. Specific optimizations are to:

- Operations on btrees that support indexes
- Add to and remove from IdentityBag and IdentitySet
- Inserts into small SequenceableCollections

The incremental logging of objects has been reduced from 1/16 to 1/32 of an object.

To reduce the space required for commits, the new configuration option GEM_COMPRESS_TRANLOG_RECORDS (page 56) allows you to write lz4 compressed tranlog records. This is true by default; the transaction logs generated in v3.4 are significantly smaller than those generated without compression.

The new configuration option STN_GROUP_COMMITS (page 58) allows you to group commits into a tranlog write.

The logsender in a hot standby system can transmit the record-level compressed transaction logs, and can now also read manually file-level compressed tranlogs (see bugs on page 113).

Multithreaded mid-cache page servers

Similarly to how page servers were multithreaded in previous releases, the mid-cache page servers are now multithreaded. Each client gem on the mid cache host will have a thread in a single mid-level cache page server.

Listening ports in for mid-cache page servers come from the pool specified by STN_PGSVR_PORT_RANGE.

Commit Record disposal without disk I/O

Using the existing configuration STN_COMMIT_RECORD_QUEUE_SIZE, commit record pages may be cached in memory, avoiding the need for disk I/O to dispose. However, bitmaps that are too big to compress onto the commit records still require disk access.

To avoid this, the configuration parameter **STN_COMMIT_RECORD_BM_CACHING** (page 58) has been added. This is true by default and allows reduced I/O during commit record disposal, most important when the commit record backlog is high or there is a lot of page preemption in the shared cache.

This setting does reduce the maximum commit rate. If your commit record backlog is usually small, and you need the maximum commit performance, set this to false.

Using multiple sizes of large memory pages on Linux

Newer Linux systems may be able to support both 2GB and 1GB large pages. Previously, it was required that the desired GemStone large page size be set as the default large page size for the system.

Now, GemStone can be configured to use either size of pages without resetting the OS default page size, using a new configuration parameter SHR_PAGE_CACHE_LARGE_MEMORY_PAGE_SIZE_MB.

This only applies to Linux. On AIX, GemStone only supports 16MB large pages (in addition to normal page sizes), so this configuration parameter does not apply. Other platforms do not require special large page configuration.

See SHR_PAGE_CACHE_LARGE_MEMORY_PAGE_SIZE_MB (page 57) for details.

1.4 Highlights of new features

Single sign-on using Kerberos

GemStone now supports Single Sign-On (SSO) using Kerberos. With SSO, logins to GemStone do not require a GemStone password; GemStone uses a current Kerberos ticket to authenticate the GemStone userId.

The NetLDI can also be configured to use Kerberos to authenticate the Unix userId, when authentications are required (that is, when the NetLDI is not in guest mode).

There are a number of ways to configure one or more UserIds for single sign-on authentication, including by group membership.

A new class, UserProfileGroup, replaces the old use of Strings as groups, and group manipulation can now be done more logically.

For details on all these changes, see “Single sign-on with Kerberos”, starting on page 33.

Secure backups

Using secure backups, you can create a backup file that is digitally signed and optionally encrypted. Creating the backup requires one or more RSA keypairs, for signing and encrypting; DSA is not supported. One or up to eight public keys is used for encryption; subsequently restoring the backup requires any one of the matching private keys.

Secure backups are created and restored using specific protocol. These methods support the same features as regular backups; multiple files, compression, etc. Secure backups are created with the extension '.sdbf'.

copydbf has added options to report and extract information from a secure backup.

For details on secure backups, see "Secure Backups", starting on page 38.

Cache warming the working set of data pages

Cache warming of data pages traditionally loaded pages by pageId, irrespective of whether the pages with lower pageIds were useful or not; this meant it was of limited value if the cache was not large enough to hold the entire repository. Now, you can configure your system to reload the specific set of data pages that was previously in the cache. See "Cache warming a working set", starting on page 60.

In addition, there is an improved way to configure cache warming to run on startup, and startstone and waitstone can wait for cache warming to complete.

Backup, restore, copydbf, and statmonitor now support LZ4

GemStone now supports LZ4 as well as gzip for backup and other compression operations. LZ4 has been used for GemStone interprocess compression for several releases, and provides much better compression performance. LZ4-compressed files have the extension .lz4 while gzip-compressed files have the .gz extension. The API has been modified to allow specification of the compression format.

See "Backup and restore support lz4 compression" on page 47, "copydbf supports lz4 compression" on page 59, and "statmonitor supports lz4 compression" on page 64.

GsBitmaps

GsBitmaps are an abstraction for the old hidden set internal bitmap structures that were accessible via System class API. The public hidden set API is now "legacy" (with the exception of a few operations that have a distinct API), and may be deprecated in a future release.

GsBitmap provides a collection-like API, but cannot include specials or temporary objects and cannot be committed. Like hidden sets, GsBitmaps are in C heap and do not use temporary object memory.

A number of operations that previously put results in hidden sets now return instances of GsBitmap.

For details, see "Bitmap based operations with new class GsBitmap", starting on page 49.

New Reduced-Conflict Collection classes

A number of new RC classes have been added that use replay to resolve certain kinds of conflict, rather than relying on session-based subcollections. These classes are more efficient, and do not require maintenance to avoid inadvertently retaining references to obsolete objects; but may avoid conflict for only a subset of collection operations.

If multiple sessions have resolvable conflicts at a rapid rate, it is possible for replay-based conflict resolution to have performance issues, since replay for the same conflict may need to be performed multiple times. In these cases, a session based subcollection structure may be more efficient.

The new reduced-conflict collection classes include:

- **RcArray**
- **GsPipe** and **RcPipe**, with FIFO queue semantics similar to RcQueue
- **RcIdentitySet**
- **RcLowMaintenanceIdentityBag**, with semantics of RcIdentityBag but with a replay rather than session-subcollection based implementation, and intended to eventually replace RcIdentityBag.

Detailed descriptions of these classes start on page 75.

Indexing changes

The indexing system has an additional internal structure that can significantly improve performance for queries. Both the new btreePlusIndex and the traditional legacyIndex internal infrastructures are available.

By default, upgraded applications will continue to use legacyIndex. It is simple to rebuild using btreePlusIndexes, but performance testing should be done, and there are element type restrictions required to get the best performance using optimizedComparison.

The type of indexing infrastructure is configured using GsIndexOptions, which has an expanded and clarified default and combination semantics. You may now set a system or session default for GsIndexOptions.

The new replay-based RC UnoderedCollections, RcIdentitySet and RcLowMaintenanceIdentityBag, only support indexes of type btreePlusIndex, not legacyIndex.

Queries can now be run on instances of collections that do not support indexes, such as Array.

Indexing changes are described under “Indexing changes”, starting on page 80.

1.5 Removed and deprecated classes and methods

Classes no longer in Globals

The class GsMethod has been moved to ObsoleteClasses.

Replaced methods

_sharedCounter:

A public interface to the `sharedCounter:*` methods was added in v3.2. In this release, the duplicate private methods have been removed.

```
System class >> _sharedCounter:incrementBy:
System class >> _sharedCounter:setValue:
```

Applications that used shared counters in v3.1.x or earlier may need to update their code.

enableGemStoneAuthentication

The method `UserProfile >> enableGemStoneAuthentication` has been removed; it is replaced by `enableGemStoneAuthenticationWithPassword:`. A password is required for GemStone authentication to be enabled.

Methods deprecated in v3.4

BinaryFloat Exceptions

With the new implementation of floating point exceptions, a number of existing methods to manage exceptions in BinaryFloat are no longer functional, but remain as deprecated. See the list under “FloatingPoint signalling Exceptions” on page 87

Finding object reference paths

v3.4 includes a new implementation of findReferencePaths, invoking a new class. The Repository methods have been deprecated. See “Finding Reference Paths” on page 46.

The implementation of findAllReferencePaths did not reliably scale, and the methods that support this have been deprecated. See the specific list of methods under “Multiple Reference Path Scans” on page 47.

Methods replaced by GsBitmap versions

GsBitmap replaces hidden sets, with an improved API the following methods are deprecated, although as they invoke primitives they do not send deprecated:

```
Repository >> auditPageOrderOopFileWithId:
Repository >> closePageOrderOopFileWithId:
Repository >> numberOfObjectsInPageOrderOopFileWithId:
Repository >> openPageOrderOopFile:
Repository >> readObjectsFromFileWithId:startingAt:upTo:into:
```

Backup methods that do not specify type of compression

Backup now supports both gz and lz4 compression. Methods to produce compressed backups that do not specify which type are deprecated; see "fullBackupCompressedTo: deprecated" on page 48

Other newly deprecated methods

```
Repository >> _pagesWithPercentFree:doScavenge:
Repository >> _validateSegmentId:
System class >> fastFindObjectsLargerThan:limit:
System class >> findObjectsLargerThan:limit:
UserProfileSet >> membersOfGroup:
```

Removed Deprecated Methods

Removed deprecated or obsolete methods

```
IcuCollator >> hiraganaQuarternary:
ProfMonitor >> spyOn:
Repository >> setArchiveLogDirectories:
Repository >> setArchiveLogDirectory:
UnorderedCollection >>
   createEqualityIndexOn:withLastElementClass:commitInterval:
UnorderedCollection >> createIdentityIndexOn:commitInterval:
UnorderedCollection >> createEqualityIndexOn:commitInterval:
UnorderedCollection >> createEqualityIndexOn:
UnorderedCollection >> createRcEqualityIndexOn:
```

Also, the methods `rehashForConversion`, `convertPoolDictionary`, and `conversionRebuild` have been deleted. These were associated with a conversion in the GemStone/S 32-bit product.

Constraints methods

Class definition-based constraints (type restrictions on instance variable values) have been deprecated since the earliest versions of GemStone/S 64 Bit.

As of v3.4, internal support for class-based constraints has been removed. The constraints field remains for documentation purposes, but many of the private and previously-deprecated methods that specifically reference or depend on constraints have been removed.

The following constraints-related methods have been removed (as well as other private constraints methods):

```
Behavior >> addInstVar:withConstraint:
Behavior >> allConstraints
Behavior >> constraintOfInstVar:
Behavior >> constraintOn:
```

```
Behavior >> elementConstraint
Behavior >> instVar:constrainTo:
Behavior >> varyingConstraint
Behavior >> varyingConstraint:
Dictionary >> constraint
UnorderedCollection >> getLastElementConstraintOnPath:
```

Removed Ruby Methods

These are replace by more general environment-based methods, see "Support for multiple execution environments" on page 74.

```
Behavior >> isRubySingletonClass
Behavior >> rubyPrimaryCopy
Behavior >> rubySuperclass:
GsNMethod >> isRubyBridgeMethod
GsNMethod >> rubyMethodProtection
GsNMethod >> rubyOptArgsBits
GsNMethod >> rubyPrivate
GsNMethod >> rubyProtected
Module >> rubyHierarchy:
Module >> rubySuperclass:
```

Removed Private Methods

The following private methods have been removed

Bag >> _addConstrained:withOccurrences:, _removeConstrained:withOccurrences:, _removeConstrainedIfPresent:withOccurrences:

Behavior >> _constraintOn:, _ivNameAndConstraintAt:, _ivOffsetAndConstraint:, _namedIvConstraintAtOffset:, _newConstraint:atOffset:, _newInheritedConstraint:atOffset:, _pathToString:, _setConstraintBit, _setVaryingConstraint:, _stringToPath:, _updateVaryingConstraint:, _validateNewInheritedConstraint:atOffset:, _validateNewVaryingConstraint:, _validateSubConstraintOf:

BinaryFloat class _exceptionFor:, _exceptionKind:, _installException:on:, _setException:to:

Class >> _checkConstraints:instVarNames:, _constraintCreationExpression, _constraintsEqual:, _equivalentSubclass:*constraints:*, _newKernelIndexableSubclass:*constraints:*, _newKernelSubclass:*constraints:*, _setClassVars:, _subclass:*constraints:*

ClassOrganizer class >> _newWithRoot: symbolList:

GsCompoundClause >> executeAndDo: using:

GsObjectInventory class>>_objInventory: waitForLock:pageBufSize: percentCpuActiveLimit:showHiddenClasses:aHiddenSet: listInstances:toFile:

GsObjectSecurityPolicy>>_migrateGroups

GsProcess>>_handleTerminateProcess:

IdentityBag>>_addAll:, _removeAll:errIfAbsent:

IndexList>>_putCommonPathTermsForPathArray:into:

Metaclass3 >> _primSubclass:*constraints:*, _setClassVars:, _subclass:*constraints:*

Module >> _setClassVars:

Object>>_changeClassTo:

PathEvaluator>>asMostSpecificType, changeEvaluatorToClass:, constraintClasses, constraintClasses:, _hasEnumeratedTerm, _hasSelectorTerm, _hasSetValuedTerm

ProcessorScheduler>> _doPoll:timeout:, _findReadyProcess, _reapEvents:, _reschedule

ProfMonitor>>_intervalString:, _invocationCountReport DownTo:, _objCreationReportDownTo:, _samplingReportDownTo:with:, _sendersReportDownTo:, _stackReportDownTo:total:

ProfMonitorTree>>_methodTreeReportDownTo:, _objectCreationTreeReportDownTo:

RcIdentityBag class >> _componentConstraint

Repository>> _findConnectedObjs, _findReferencePathToObjs:limitObjArray:findAllRefs:printToLog:, _fullBackupTo: MBytes:compress:, _listReferencesInMemory:, _primLoadGcCandidatesFromFile: intoHiddenSet , _restoreBackups: scavPercentFree:

System class>>waitForRcWriteLock:, _acquireRcLock, _findNonDestructiveInRangeFrom: to:startingAt:to:, _inMap:at:putIfAbsent:, _lockGc:, _rcLockObject, _releaseRcLock, __commit:

UndefinedObject >> _newKernelSubclass:*constraints:*

UnorderedCollection >> _getConstraintsOnPath:, _getLastElementConstraintOnPath:onError:, _isLastOccurrenceInInde xObjectsFor:, _removeSubsumedIndex:

UserProfile>>_migrateGroups

Chapter

2 Changes in Administration

This chapter describes changes and new features that apply to the administration of a GemStone Repository, including:

2.1 UserProfile and groups changes

Enable and disable

Previously, disabling of UserProfile was bound to the GemStone authentication scheme, and had some logically inconsistent behavior. It was not supported to manually disable accounts using non-GemStone authentication. These processes have been reviewed and the code updated.

Now: any UserProfile can be explicitly disabled using `disable` or `disableWithReason:`. UserProfiles that use GemStone authentication can be re-enabled using:

```
reenableWithPassword:
password:
```

UserProfiles not using GemStone authentication that were explicitly disabled can be re-enabled using:

```
reenable
```

GemStone authentication, but not external authentication, supports automatic disable for login checks, such as exceeding STN_DISABLE_LOGIN_FAILURE_LIMIT. Accounts disabled in this way are also reenabled using `reenableWithPassword:` or `password:`.

When using external (non-GemStone) authentication, password security login checks are handled by the authentication mechanism, and no disable is done in GemStone. If logins are disabled by the system that performs the authentication, you must re-enable the account as defined by that system, outside of GemStone.

For example, if using LDAP authentication and the LDAP account is disabled for too many failed logins, the GemStone UserProfile is not "disabled", although logins will fail. After correcting the problem with the LDAP account, you do not need to re-enable the GemStone UserProfile.

enableGemStoneAuthentication requires password argument

The method `UserProfile>>enableGemStoneAuthenticationWithPassword` has been removed, replaced by `UserProfile>>enableGemStoneAuthenticationWithPassword:`

Since enabling GemStone authentication requires a password, the earlier version was not sufficient.

Bypass GsCurrentSession initialize in topaz

On login, the method `GsCurrentSession >> initialize` is invoked, which in turn does session method initialization and invokes that UserProfile's loginHook:, if present. In much older releases before loginHook was introduced, the `GsCurrentSession >> initialize` method could be modified. Errors or problems in these functions, however, can prevent logins from completing.

To allow logins under these error conditions, the topaz option SET SESSIONINIT has been added, further controlled by the new configuration parameter STN_ALLOW_NO_SESSION_INIT. For details, see "SESSIONINIT to allow login when session initialization fails" on page 63.

UserIds outside the ASCII range

It has been possible to create userIds (user names) containing characters in the range 128..255. However, the topaz command line input is always UTF-8, and the internal storage of userIds is traditional string, and as a result logins on the topaz command line were not possible. Now, you may enter such userIds at the topaz command line. (#46370)

It is not legal to use characters with codePoints over 255 in userIds or passwords; this has been disallowed to avoid unusable UserProfiles.

UserProfile empty passwords disallowed

Previously, an empty string was a legal password, although not necessarily usable for login. This has been disallowed; the minimum password size is now 1. A password size of zero indicates single-sign on.

Code that verifies passwords has also been cleaned up.

UserProfile Instance variable slot for spare1 now reused

The UserProfile slot used by instance variable `spare1` is now `kerberosPrincipal`.

UserProfileGroup replaces string-based groups

Historically, a Group has been represented by an instance of String with the name of the group, and these group strings were added to a UserProfiles's groups variable, as well as to the Global AllGroups.

To support group-based single sign-on, in v3.4 this implementation has been changed, and a group is now an instance of the new class UserProfileGroup. AllGroups now maps group names to instances of UserProfileGroup, and each UserProfile's group field contains a set of UserProfileGroups.

During upgrade, existing groups are automatically converted.

Creating and Deleting Groups using UserProfileGroup

To create a group and add it to all groups, you no longer add a String directly to AllGroups. The new process is to create the group using UserProfileGroup class methods. For example,

```
UserProfileGroup newGroupWithName: 'finance'
```

To lookup a group based on its string name, also use UserProfileGroup class methods. For example,

```
UserProfileGroup groupWithName: 'finance' ifAbsent:
  [UserProfileGroup newGroupWithName: 'finance']
```

Removing a group likewise uses UserProfileGroup class methods `deleteGroup:*`.

The AllGroups global should not be accessed directly when querying or updating groups. Use the class methods in UserProfileGroup instead.

Connecting UserProfiles and Groups

Existing UserProfile methods such as `addGroup:` and `removeGroup:` that manage group membership continue to accept group string arguments; these methods perform a lookup to UserProfileGroups as needed.

New methods have been added to add groups to and remove them from a UserProfile:

```
UserProfile >> addToUserProfileGroup:
UserProfile >> removeFromUserProfileGroup:
```

In addition to the traditional API in which groups were added to users, protocol has been added to allow users to be added to groups, which may be more intuitive.

```
UserProfileGroup >> addUser:
UserProfileGroup >> removeUser:
```

Membership

To find the users in a group, use:

```
UserProfileGroup >> userIds
```

Return a set of all UserId strings for each member of the group

`UserProfileGroup >> users`
Return a set of UserProfiles that includes each member of the group

To find the groups for a user use:

`UserProfile >> groupNames`
Return a set of all group strings for each group that the given user belongs to.

`UserProfile >> groups`
Return a set of UserProfileGroups that includes each group that the given user belongs to.
Note: in previous versions, this method returned a set of Strings rather than a set of UserProfileGroups.

Group audit

Methods have been added to support an audit of group membership.

`UserProfileGroup >> auditGroup`
Verify that all users in the receiver do have the receiver in their groups array.

`UserProfileGroup class >> auditGroups`
Audit each group in AllGroups.

`UserProfile >> auditGroups`
Verify that each group associated with this UserProfile does include this user in its members, and that each group is in AllGroups.

`UserProfileSet >> auditGroups`
Audit groups for each user in AllUsers (the singleton instance of UserProfileSet), and audit UserProfileGroup.

2.2 Single sign-on with Kerberos

GemStone now supports Single Sign-On (SSO), using Kerberos. When Kerberos has been setup on your system, you can configure GemStone authentication to use Kerberos, so no separate authentication is required for GemStone login. This may also be referred to as passwordless login.

Note that while most users can use SSO authentication, SystemUser will always require GemStone authentication. Since SSO users cannot log in if Kerberos is not running, this avoids the risk of complete inaccessibility.

Both linked and RPC logins and remote logins, including logins from GBS and using GsExternalSession, can use SSO to authenticate, provided the user can authenticate via Kerberos on the client's node.

Kerberos

Using SSO with Kerberos requires that Kerberos be installed and configured on your system. For specifics, consult your System Administrators. The general requirements are outlined here.

GemStone uses Kerberos v5 Release 1.15.2 with AES encryption.

Realm

Kerberos uses the term "realm" for a domain within an installation/organization. Realms are generally the DNS domain in upper case; for example, at GemTalk the realm would be GEMTALKSYSTEMS.COM.

User Principals

The term "principal" is used for any unique identity, including UNIX user ids, hosts, and services.

User principals are identified as name@REALM, e.g.

```
lalmarod@GEMTALKSYSTEMS.COM
```

When a user authenticates using Kerberos (e.g by UNIX login or by using kinit), the user enters their password, and Kerberos provides a ticket -- more specifically, a ticket granting ticket (TGT). This is cached on the user's host, and is used to create specific tickets for requested services at the time they are requested. The TGT is usually put under /tmp, and is valid for a limited period, by default usually 10 hours.

GemStone Service Principals

In addition to the user principals, you will need to have service principals for GemStone. The service principal is of the form Service/fully.qualified.domain.name@REALM. The name of the service principal for authentication in GemStone is GEMSTONE64. So, for example, a service principal might be

```
GEMSTONE64/santiam.gemtalksystem.com@GEMTALKSYSTEMS.COM
```

Keytab

A keytab is a file containing pairs of Kerberos principals and encrypted keys. GemStone requires a keytab that stores keys for the GEMSTONE64 kerberos service, which must be there for each host. This keytab does not store user principals.

This keytab file will be used for authorization; any user who has authenticated via Kerberos and has access to the keytab file will be able to do a single sign-on login.

KerberosPrincipal and AllKerberosPrincipals

The new class KerberosPrincipal defines the association between the name of a Kerberos principal and GemStone login information. A KerberosPrincipal has the following information:

`name`	(Symbol) the name for the kerberos principal; must be unique.
`loginUserProfile`	a UserProfile, or nil for a shared KerberosPrincipal.
`loginUserProfileGroups`	an IdentitySet of UserProfileGroups, to support a shared KerberosPrincipal, or nil.
`loginAsAnyoneEnabled`	a Boolean; true if any UserProfile can use this KerberosPrincipal.

There are a number of ways that a repository can be configured to allow one or more UserProfiles to login using one or more KerberosPrincipals.

- There may be a one-to-one relationship between a UserProfile and a KerberosPrincipal, with each referencing the other. In this case `loginUserProfileGroups` and `loginAsAnyoneEnabled` are usually nil.
- One or more UserProfile/s may reference a KerberosPrincipal with `loginUserProfileGroups` configured as a set of UserProfileGroups. The KerberosPrincipal has a nil `loginUserProfile` and `loginAsAnyoneEnabled` set to false. Any user that is a member of any of these groups can use this KerberosPrincipal to log in.
- One or more UserProfile/s may reference a KerberosPrincipal that has a nil `loginUserProfile`, and with `loginAsAnyoneEnabled` set to true. This allows any user to configure SingleSignOn using that KerberosPrincipal.

KerberosPrincipals are created using class protocol in KerberosPrincipal, and are automatically added to the global collection AllKerberosPrincipals. You can look up a KerberosPrincipal by name. The AllKerberosPrincipals global should not be accessed directly.

The login log includes the Kerberos principal (page 52), and descriptionOfSession: results includes Kerberos principal (page 52).

UserProfile new methods

Methods that set or query for the authentication scheme can now use #SingleSignOn in addition to #GemStone, #UNIX, or #LDAP.

To setup singleSignOn authentication, use

`enableSingleSignOnAuthenticationWithPrincipal:` *aKerberosPrincipal*

To query if this authentication is enabled use:

```
authenticationSchemeIsSingleSignOn
```

To query and set the KerberosPrincipal for a particular user use:

```
kerberosPrincipal
kerberosPrincipal: aKerberosPrincipal
```

Setting up Kerberos Authentication in GemStone

Within GemStone, the following steps are required:

1. Create or locate the UserProfile

The UserProfile who will use SSO will have their authentication changed, and will no longer use the GemStone password. This user will only be able to login as long as Kerberos is running. For this reason, the SystemUser account cannot be set to SSO.

2. Create an instance of KerberosPrincipal

A UserProfile who will use SSO is associated with an instance of the new class KerberosPrincipal.

3. Enable SSO for that UserProfile

Single sign-on/Kerberos is another way to do GemStone authentication, enabled via `enableSingleSignOnAuthenticationWithPrincipal:`.

You may disable SSO by enabling a different type of authentication.

4. Commit the changes to the UserProfile

As an example of the previous steps, the following code configures the user named 'john' to use SSO:

```
| prin user |
user := AllUsers userWithId: 'john'.
prin := KerberosPrincipal
      newPrincipalWithName: 'jsmith@GEMTALKSYSTEMS.COM'
      loginUserProfile: user.
user enableSingleSignOnAuthenticationWithPrincipal: prin.
System commitTransaction.
```

5. Configure the gem's environment to set the keytab file location

The Gem configuration parameter GEM_KERBEROS_KEYTAB_FILE must be set in order to specify the location of the keytab file.

For example, enter this in the configuration file used by john:

```
GEM_KERBEROS_KEYTAB_FILE =
/export/localnew/common/kerberos/gemtalk.keytab;
```

6. Login without password

On GemStone login, do not set the password (leave as an empty string).

The login will use the KerberosPrincipal's name to lookup the Kerberos ticket and authenticate the user, and the keytab, to verify access to the GemStone server.

Using Groups to authenticate with Kerberos

A KerberosPrincipal often will correspond to a specific GemStone UserProfile. However, you may use groups to configure your system so multiple GemStone UserProfiles can login using the same KerberosPrincipal.

To do this, create a KerberosPrincipal with a nil loginUserProfile. Create a new group and make the users part of the group, and add the group to the KerberosPrincipal. You can then use this KerberosPrincipal to enable single sign-on for all the users in the group

For example, the following code uses jsmith's principal to login as both 'john' and 'mary'.

```
| prin user1 user2 group |
user1 := AllUsers userWithId: 'john'.
user2 := AllUsers userWithId: 'mary'.
group := UserProfileGroup
      newGroupWithName: 'KerberosAdminLogin'.
user1 addGroup: group groupName.
user2 addGroup: group groupName.
prin := KerberosPrincipal newPrincipalWithName:
      'jsmith@GEMTALKSYSTEMS.COM' loginUserProfile: nil.
prin addGroup: group.
group users do: [:aUserProfile | aUserProfile
      enableSingleSignOnAuthenticationWithPrincipal: prin].
System commitTransaction.
```

You can configure all users on your system (except SystemUser) to share the KerberosPrincipal by using

aKerberosPrincipal `loginAsAnyoneEnabled: true`

This is equivalent to creating a group that includes all users except SystemUser and using this for the KerberosPrincipal's group.

Then, any user (again, except SystemUser) can set their authentication to #SingleSignOn using the principal *aKerberosPrincipal*.

The AllKerberosPrincipals global should not be accessed directly when querying or updating KerberosPrinciples. Use the class methods in KerberosPrincipals instead.

Remote login on Windows

Single sign-on can be used to login clients on Windows. On Windows, Kerberos is built-in as part of SSPI.

Single sign-on can only be used on Windows clients that are part of a Windows domain; clients on machines that are only members of Windows workgroups cannot use Kerberos authentication. The domain controller must either be samba or a MS Windows server.

Consult your system administrators or GemTalk Technical Support for clarification or assistance.

GemBuilder for Smalltalk

Single sign-on can be used for logins from GBS clients on Windows and Unix.

However, given the various ways KerberosPrincipals can be configured, the GBS User Tools do not support configuring single sign-on interactively. You must setup KerberosPrincipals and assign to users using GemStone code, as described above.

Login log includes Kerberos principal

When STN_LOGIN_LOG_ENABLED is set, the login log entry that is written now includes another field within each line. The new field, at the end of each line, lists the name of the KerberosPrincipal used for single sign-on login enclosed in single quotes, or '' (two single quotes) if single sign-on login was not used.

descriptionOfSession:

The result of `System descriptionOfSession:` now includes an additional element (22): The KerberosPrincipal object used for single sign-on login to the session, or nil if single sign-on login was not used.

Using Kerberos for NetLDI authentication

The NetLDI can be configured to use Kerberos for authentication for the hostUserId/hostPassword. A new argument as been added to startnetldi to allow a Kerberos keytab to be specified:

```
-k <keytabPath>
```

This option can only be applied when NetLDI requires authentication, since in guest mode, no authentication is done. It is disallowed to specify both -g and -k.

To configure Kerberos authentication for NetLDI:

- startnetldi must be executed with the -k option argument of the Kerberos keytab file path.
- The GemStone login parameters must have an empty hostPassword field, and the username must match the one in the Kerberos ticket.
- There must be a currently valid ticket (TGT) associated with a principal for the UNIX user id that the NetLDI is authenticating as.

2.3 Secure Backups

The new secure backups feature provides two new capabilities:

- Digital signing, ensuring the identity of the originator. All backups created using the secure backup API are digitally signed.
- Encrypted backups, requiring a private key to restore. Encryption is optional, using AES-CTR with key sizes of 128, 192, or 256.

Secure backups, as with standard backups, can also be written to one or multiple files and may be uncompressed, compressed using gzip, or compressed using lz4.

Secure backups are created using a new Repository method that contains a number of arguments to specify the filenames, encryption type, and other details of the backup.

```
secureFullBackupTo:MBytes:compressKind:bufSize:encryptKind:
   publicKeyCerts:signatureHashKind:signingKey:
   signingKeyPassphrase:
```

Restoring a secure backup uses new Repository methods

```
restoreFromSecureBackup[s]:*.
```

You cannot use the secure restore to restore a regular backup, nor vice versa.

Secure backups require RSA keypairs, both for signing and encryption (if used); DSA is not supported. You will need separate keypairs for signing and encryption. The signing key may or may not require a passphrase, although a passphrase is recommended for security. The example keys included in the distribution require passphrases.

To create a secure encrypted backup, you can use one or up to eight public keys. Restoring this backup requires a private key that corresponds to any one of these public keys. This allows you to create a backup that can be restored by multiple entities, without these entities having to share a single private key.

Before using the secure backup API to make or restore secure backups, you must set the keyfile directories (keyring), using the new Gem configuration option GEM_KEYRING_DIRS. This can be set either in the configuration file used by the Gem process, or set at runtime. The directories specified in GEM_KEYRING_DIRS are used to confirm there is a matching public keyfile for keys used by the secure backup operations, and that private keys match the public keys.

Example keys and passphrases

The directory `$GEMSTONE/examples/openssl` includes two subdirectories with example keys for creating, signing, and restoring secure backups:

- `certs` contains public keys
- `private` contains private keys and passphrases

These keys are used in the examples starting on page 42.

Added Methods

Making the Backup

`Repository >> secureFullBackupTo:` *fileNames* `MBytes:` *mByteLimit*
`compressKind:` *compKind* `bufSize:` *count* `encryptKind:` *encKind*
`publicKeyCerts:` *anArrayOrString* `signatureHashKind:` *hashKind*
`signingKey:` *signingKeyFn* `signingKeyPassphrase:` *aPassphrase*

Writes a secure full backup file containing the most recently committed version of the receiver as of the time the method is executed.

Secure backups support compression, encryption and digital signatures. Compression and encryption are optional. Digital signatures are mandatory.

fileNames must either be a string which is the backup file name or an array of strings containing a list of backup file names.

Secure backups have a file extension of .sdbf, which will be appended to all backup file names if necessary.

compKind argument:

The following compression kinds are supported. LZ4 is recommended since it is faster than zlib.
0 - no compression
1 - zlib (aka gzip) compression.
2 - lz4 compression.

encKind argument:

An *encKind* of zero indicates the backup is not encrypted. An *encKind* greater than zero indicates the backup is encrypted using AES-CTR mode using the key size shown below:
0 - no encryption
1 - AES-CTR-128
2 - AES-CTR-192
3 - AES-CTR-256

hashKind argument:

hashKind specifies what message digest (hash) algorithm will be used to hash the signature before it is encrypted and stored in the backup file. The following message digests are supported:
1 - SHA256
2 - SHA384
3 - SHA512

All certificates and keys must be in one of the directories specified in the key ring, which is the list of directories specified in the GEM_KEYRING_DIRS configuration option. Certificate and key file names must be specified without the path. GemStone will automatically search the key ring for the matching file.

If *encKind* is greater than zero, the `publicKeyCerts` argument *anArrayOrString* must either be a String or an Array of Strings where each element is the filename of an X509 certificate in PEM format that contains an RSA public key without a passphrase. The RSA public key(s) may be any valid length (1024, 2048, 3072, bits etc). A key length of at least 2048 bits is recommended. *anArrayOrString* must be nil if *encKind* is zero. A maximum of eight (8) public key certificates may be specified.

When encryption is enabled, each backup file is encrypted with a randomly generated session key, which is then encrypted with the provided public key(s) and stored in the backup file. One of the corresponding private keys must be available in order to restore the backup, but are not required when the backup is created.

signingKeyFn is the name of a private key file in PEM format used to sign the backup. The file must be in the key ring. *aPassphrase* is a string which is the passphrase for the private key. If the private key does not use a passphrase, then *aPassphrase* must be nil. A certificate containing the public key which matches the private key specified in privateKeyFn is stored in the backup and must also be present in the key ring. GemStone automatically searches the keyring for the matching certificate.

The *bufSize* argument controls the size of the buffer used to write records to the file. The count argument specifies the number of 128KB backup records are contained in the buffer. The values allowed are 1 (128KB) through 8192 (1GB).

Restoring the Backup

`Repository >> restoreFromSecureBackup:` *aFileName*
`privateDecryptionKey:` *aKey* `passphrase:` *aPassphrase*

`Repository >> restoreFromSecureBackup:` *aFileName*
`scavengePagesWithPercentFree:` *aPercent* `privateDecryptionKey:` *aKey*
`passphrase:` *aPassphrase*

`Repository >> restoreFromSecureBackups:` *anArrayOfFilenames*
`privateDecryptionKey:` *aKey* `passphrase:` *aPassphrase*

`Repository >> restoreFromSecureBackups:` *anArrayOfFilenames*
`scavengePagesWithPercentFree:` *aPercent* `privateDecryptionKey:` *aKey*
`passphrase:` *aPassphrase*

Disables logins and starts a full restore of the repository based on the contents of the specified secure backup file(s). Behaves the same as the `restoreFromBackup:` method except for extra security and integrity checks as described below.

Secure backup files must have a file extension of .sdbf. All file names in *arrayOfFileNames* will have the .sdbf suffix appended if it is not present.

All certificates and keys must be in one of the directories specified in the key ring, which is the list of directories specified in the GEM_KEYRING_DIRS configuration option. Certificate and key file names must be specified without the path. GemStone will automatically search the key ring for the matching file.

Secure backups are always digitally signed with a private signing key, and optionally were encrypted using one or more public keys. Before the backup is restored, the integrity of each backup file is checked by reading the file and computing the signature. If the computed signature does not match the signature stored in the file, then an error is raised indicating the backup file has been either been corrupted or tampered with. If the signature is correct, the restore proceeds.

A certificate containing the public key which matches the private key used to sign the backup must be present in the key ring. GemStone automatically searches all files in the key ring for a certificate which contains a matching public key. The restore will fail and an error will be raised if the key ring does not contain the appropriate certificate.

If the backup was encrypted, *aKey* is a String representing the file name containing a private key in PEM format. The private key must match one of the public keys

specified when creating the backup. The file must be present in the key ring as described above. If the backup is not encrypted then *aKey* must be nil.

aPassphrase is the passphrase to access the private key. If the backup is not encrypted or *aKey* is not protected by a passphrase (not recommended) then *aPassphrase* must be nil.

This method performs the restore using multiple slave sessions. The number of slave sessions is automatically determined from the number of extents in the repository, with a minimum of 2 sessions and a maximum of 16. The performance can be modified during the run by updating the Multithreaded Scan Tuning methods.

Configuration option

A new configuration file option allows you to specify the location of keys and certificates used for secure backup/restore.

GEM_KEYRING_DIRS

A list of directories which contain keys and certificates used for secure backup and restore.

Runtime equivalent: #GemKeyRingDirs
Default: NONE

copydbf information about secure backups

copydbf can be used with secure backups similarly as with ordinary backup files; however, byte-order changes and copying over the network to another page server are not allowed with secure backups.

The reports produced by copydbf -i/-I include basic security attributes for secure backup files.

In addition, new copydbf arguments can be used to extract certain elements of security information from the backup.

copydbf -i now includes security information when run against a secure backup:

```
host> copydbf -i tsb.multifile.backup.bak_4.sdbf

Source file: tsb.multifile.backup.bak_4.sdbf
   File type: secure backup  fileId: 3
in a backup set with 4 files
   ByteOrder: Intel (LSB first)  compatibilityLevel: 850
   The file was created at: 09/08/17 13:28:36 PDT
   Second or subsquent file of a multi-file full backup.
   Oldest tranlog needed for restore not available.
   Backup was created by GemStone Version: 3.4.0.
   Secure Backup Attributes:
         Compression: Zlib
          Encryption: AES_128_CTR
      Signature Hash: SHA-256
     Encryption Keys: 1
```

New options have been added to extract information, and verify a secure backup file. These options allow you to extract public certificate information from the backup to a disk file. While this is provided for convenience, this information does not contribute to the security of backup administration.

The added copydbf options are:

-K *directories*
list of colon-separated directories which contain keys and certificates, which is used with the -V option to verify the digital signature.

-V verify the digital signature of a secure backup. This requires including either the -K command line option, or setting the GEMSTONE_KEYRING_TABS environment variable.

-W *filename*
Create a new file or files with the given name containing the encryption certificate(s) from a secure backup to file(s). Cannot be used with other write options on a single command line.

-X *filename*
Create a new file with the given name containing the signing certificate from a secure backup to file. Cannot be used with other write options on a single command line.

-Y *filename*
Create a new file with the given name containing the digital signature from a secure backup to file. Cannot be used with other write options on a single command line.

Examples of Secure Backup

Create a signed, unencrypted backup

The following code creates a single file secure backup, uncompressed and not encrypted, using the example key and passphrase in the GemStone distribution.

```
topaz 1> run
System gemConfigurationAt: #GemKeyRingDirs
      put: {'$GEMSTONE/examples/openssl/certs' .
      '$GEMSTONE/examples/openssl/private' }.
SystemRepository
      secureFullBackupTo: '$GEMSTONE/data/backup'
      MBytes: 0
      compressKind: 0
      bufSize: 8
      encryptKind: 0
      publicKeyCerts: nil
      signatureHashKind: 1
      signingKey: 'backup_sign_2_clientkey.pem'
      signingKeyPassphrase: (GsSecureSocket
          getPasswordFromFile:
         '$GEMSTONE/examples/openssl/private/backup_sign_2_clie
nt_passwd.txt' ).
%
```

copydbf of a signed, unencrypted backup

```
unixprompt> copydbf -i backup.sdbf
Source file: backup.sdbf
   File type: secure backup  fileId: 0
in a backup set with 1 files
   ByteOrder: Intel (LSB first)  compatibilityLevel: 850
   The file was created at: 09/08/2017 14:16:39 PDT
   Full backup started from checkpoint at: 09/08/2017 14:16:38
PDT.
   Oldest tranlog needed for restore is fileId 1 ( tranlog1.dbf
).
   Backup was created by GemStone Version: 3.4.0.
   Secure Backup Attributes:
         Compression: NONE
          Encryption: NONE
      Signature Hash: SHA-256
     Encryption Keys: 0
```

Restore an unencrypted secure backup

```
topaz 1> run
System gemConfigurationAt: #GemKeyRingDirs
      put: {'$GEMSTONE/examples/openssl/certs' .
      '$GEMSTONE/examples/openssl/private' }.
SystemRepository
      restoreFromSecureBackup: 'backup.sdbf'
      privateDecryptionKey: nil
      passphrase: nil.
%
```

Create an encrypted backup

To encrypt a backup, you can specify between 1 and 8 public certificates. To restore an encrypted backup, any one of the private keys associated with the public keys can be used. The private key is not required to create the encrypted backup, only to restore it.

For example, if an organization wishes to distribute a backup to multiple customers via an encrypted backup, the public key for each customer can be specified when the backup is created. Each customer may use their individual private key to restore the backup.

```
topaz 1> run
System gemConfigurationAt: #GemKeyRingDirs
      put: {'$GEMSTONE/examples/openssl/certs' .
      '$GEMSTONE/examples/openssl/private' }.
SystemRepository
      secureFullBackupTo: '$GEMSTONE/data/backup'
      MBytes: 0
      compressKind: 2
      bufSize: 8
      encryptKind: 2
      publicKeyCerts: { 'backup_encrypt_1_clientcert.pem' .
      'backup_encrypt_2_clientcert.pem'}
      signatureHashKind: 1
      signingKey: 'backup_sign_1_clientkey.pem'
      signingKeyPassphrase:
            (GsSecureSocket getPasswordFromFile:
'$GEMSTONE/examples/openssl/private/backup_sign_1_client_passwd
.txt' )
```

Copydbf of an encrypted backup

```
unixprompt> copydbf -i backup.sdbf
Source file: back1.bak.sdbf
   File type: secure backup  fileId: 0
in a backup set with 1 files
   ByteOrder: Intel (LSB first)  compatibilityLevel: 850
   The file was created at: 09/08/2017 14:35:10 PDT
   Full backup started from checkpoint at: 09/08/2017 14:35:09
PDT.
   Oldest tranlog needed for restore is fileId 1 ( tranlog1.dbf ).
   Backup was created by GemStone Version: 3.4.0.
   Secure Backup Attributes:
         Compression: LZ4
          Encryption: AES_192_CTR
      Signature Hash: SHA-256
     Encryption Keys: 2
```

Restore an encrypted backup

```
topaz 1> run
System gemConfigurationAt: #GemKeyRingDirs
      put: {'$GEMSTONE/examples/openssl/certs' .
      '$GEMSTONE/examples/openssl/private' }.
SystemRepository
      restoreFromSecureBackup: 'backup.sdbf'
      privateDecryptionKey: 'backup_encrypt_1_clientkey.pem'
      passphrase: (GsSecureSocket getPasswordFromFile:
'$GEMSTONE/examples/openssl/private/backup_encrypt_1_client_pas
swd.txt' )
%
```

Script to support verifying digital signature

A script has been added that uses publicly available openSSL, by default the executable in $GEMSTONE/bin/openssl, to verify the digital signature of a secure backup.

```
$GEMSTONE/bin/verify_backup_with_openssl.sh
```

Example

Verify the signature using OpenSSL:

```
% $GEMSTONE/bin/verify_backup_with_openssl.sh backup.sdbf
   $GEMSTONE/examples/openssl/certs/backup_sign_2_clientcert.pem
[Info]: Using certificate file
   /lark1/GS/examples/openssl/certs/backup_sign_2_clientcert.pem
[Info]: Digest kind is SHA-256
[Info]: Invoking openssl to perform the digital signature
   verification:

Verified OK
```

Copydbf can also be used:

```
% copydbf -V backup1.sdbf -K $GEMSTONE/examples/openssl/certs/
Source file: backup1.sdbf
   File type: secure backup  fileId: 0
in a backup set with 1 files
   ByteOrder: Intel (LSB first)  compatibilityLevel: 850
   The file was created at: 09/07/2017 23:26:16 PDT
[Info]: Digital signature successfully verified.
```

2.4 Changes related to Repository Scan operations

Repository scan operations are those that need to scan the entire repository, including backup creation, garbage collection, audit, and operations that analyze the objects in a repository.

Concurrent Repository Scan Operations

Historically, only one repository scan operation could be run at one time within a given GemStone repository, to avoid the risk of conflicting updates. Repository scan operations include garbage collection (MFC, FDC, and Epoch), backups, object audit, and listing operations such as list instances and list references. The single gcLock limited reclaim operations, so only pages that have been scanned (below the gcHighWaterMark) were allowed to be processed for reclaim.

While two garbage collection operations cannot be safely run simultaneously, with v3.4, it is now possible to run one garbage collection operation and up to three other repository scans at the same time within a single GemStone repository, provided the repository's I/O and CPU can support the load.

Note that if there is more than one scan performed simultaneously, shadow page reclaim is disabled for the duration. Only dead objects will be reclaimed.

Object Profiling of objects in temporary object memory

GsObjectInventory can now create a report on the contents of temporary object memory, with the added method:

```
GsObjectInventory class >> profileMemory
```

Finding Reference Paths

A new subsystem has been added to provide optimized, multi-threaded repository scans for reference paths. This subsystem replaces the internal code invoked by the `Repository >> findReferencePath...` methods.

To perform the scan, you create an instance of the added class GsSingleRefPathFinder for the object or objects, run the scan, and collect/view the results.

GsSingleRefPathFinder provides a number of instance variables to parameterize the search:

- maxThreads
- lockWaitTime
- pageBufferSize
- percentCpuLimit
- maxLimitSetDescendantObjs
- maxLimitSetDescendantLevels
- printToLog

Defaults are provided, or you may send messages as needed, for example to perform a less aggressive scan.

To perform a simple scan and print the result string:

```
(GsSingleRefPathFinder newForSearchObjects: { anObject })
   scanAndReport
```

To perform a scan with a specific limit set on descendents:

```
inst := GsSingleRefPathFinder newForSearchObjects: { anObject }.
inst maxLimitSetDescendantLevels: 4.
inst runScan.
inst buildResultObjects collect:
      [:refPath | refPath resultString].
```

The method `buildResultObject` returns a collection of instances of the new class GsSingleRefPathResult. This class is a subclass of Array, with each element representing an element in the reference path, and instance variables for the searchOop and status.

Note that using a smaller `maxLimitSetDescendantLevels` may not be performant; if the search objects are found within the limit set, the repository scan is not needed.

Deprecated methods

The following methods on Repository are deprecated; they invoke the new classes to perform the scan:

```
Repository >> findReferencePathToObject:

Repository >> findReferencePathToObjects:findAllRefs:
   printToLog:

Repository >> findReferencePathToObjs:limitObjArray:
   findAllRefs:printToLog:
```

Note that these methods now return an instance or instances of GsSingleRefPathResult, and arguments other than the search object/s are not used.

Multiple Reference Path Scans

The functionality provided by recently-introduced queries `findAllReferencePaths:*` has been deprecated in this release; these operations were fragile when scaled to production-sized use cases.

Other added Classes

The following related classes have been added, used internally in this release and designed for improving tools for repository object analysis.

```
GsReferencePath

GsReferencePathParentsInfo

GsSingleRefPathFinderForObject
```

Backup and restore support lz4 compression

GemStone now supports lz4 for most operations that allow compression.

To support lz4 compression of programmatic backups, new API methods have been added, and restore methods now recognize and can decompress both formats.

It is expected that .gz and .lz4 extensions reliably indicate the compression used.

For backups, lz4 is generally much faster than gzip (usually by 2.5X but up to 10X) but does not compress data quite as well.

The following methods have been added to create lz4-compressed backups:

```
fullBackupLz4CompressedTo:
fullBackupLz4CompressedTo:MBytes:
```

To differentiate creating backups that are compressed using gzip from compression using lz4, new methods have been created that include "Gz" in the name:

```
fullBackupGzCompressedTo:
fullBackupGzCompressedTo:MBytes:
```

A method has also been added to programmatically specify the type of compression:

```
fullBackupTo:MBytes:compressKind:bufSize:
```

fullBackupCompressedTo: deprecated

The existing methods invoke the Gz versions, and are deprecated in v3.4:

```
fullBackupCompressedTo:
fullBackupCompressedTo:MBytes:
```

Restore of backups and transaction logs

Restore from backup and restore from logs now recognize backups and transaction log files with the .lz4 file extension and treat them as compressed lz4 files.

Buffer size

The default number of buffers for lz4-compressed backups is 16 (2 MB) vs. 1 (128k) for gz-compressed backups.

2.5 Bitmap based operations with new class GsBitmap

GsBitmap

The class GsBitmap has been added, encapsulating behavior of GemStone's hidden sets.

GsBitmap represents an in memory bitmap, i.e. a transient bitmap that cannot be made persistent. A bitmap is a sparse data structure that logically implements a bit array. Each bit in the bitmap represents the oopNumber of a committed non-special object.

GsBitmaps return an error on an attempt to add a temporary object or a special, or if you attempt to commit a reference to them. As with hidden sets, you should use caution in working with GsBitmaps, since the objects are referenced as OOPs, and if not otherwise referenced, are not protected from garbage collection nor the OOP from reuse.

While GsBitmap can be considered as a collection and implements Collection protocol, it does not inherit from Collection. You may send `asGsBitmap` to create a GsBitmap from a collection, provided the collection only contains objects that are allowed in a GsBitmap; use `asArray` to collect the objects corresponding to the OOPs in the GsBitmap.

An instance of GsBitmap uses C Heap memory to store the bit array. The C Heap memory associated with an instance of GsBitmap is automatically freed when the instance is garbage collected.

Bitmap Files

In addition to standard collection protocol, GsBitmaps can be written to and read from disk, using the following methods:

```
GsBitmap >> writeToFile:
GsBitmap >> writeToFileInPageOrder:
GsBitmap >> readFromFile:
GsBitmap >> readFromFile:withLimit:startingAt:
```

You may also query for information on a given bitmap file, using `GsBitmap >> fileInfo:`. This method returns an array containing:

- number of oops in the file
- whether the file was written in page order
- number of valid oops
- number of oops that are not allocated, or in the process of being garbage collected

File format

GsBitmap files are written in a new file format, which is not compatible with files written using hidden set protocol. You may not read files using the GsBitmap interface that were generated using the hidden set interface, and vice versa.

`findDisconnectedObjectsAndWriteToFile:*` and `loadGcCandidatesFromFile:*` also now use the GsBitmap format, rather than the previous specialized format. You may use GsBitmap methods to load the disconnected object files generated by `findDisconnectedObjectsAndWriteToFile*` methods.

Garbage Collection consequences

The objects referenced in a GsBitmap are not protected from garbage collection.

The reference within the GsBitmap is to the OOP. For a GsBitmap that exists in a session for time periods that include commits or aborts, if the object is otherwise unreferenced, there is a risk that it may be garbage collected and the OOP may be reused. Subsequently, the GsBitmap may not contain objects that were expected to be there, or may contain objects that were not expected to be there.

GsBitmap and Hidden Sets

System's hidden set API has been superceded by GsBitmap. The Hidden set public API is not deprecated, but it is marked as Legacy and may be deprecated in the future.

GsBitmap uses hidden set specifiers that are symbols, rather than integers. The method `GsBitmap class >> hiddenSetSpecifiers` lists the symbolic names for the various hidden sets.

NOTE: While the position of the symbolicName in the list of hiddenSetSpecifiers is internally used as an index, these indexes do NOT match the integers used as specifiers for System HiddenSets. Nor do the symbolic names match the strings returned by System HiddenSetSpecifiers.

Public access to most System hiddenSets is provided by creating a GsBitmap that references a particular hiddenSet, using `GsBitmap class >> newForHiddenSet:`.

Many of the System hiddenSets are read only, and generate an error for update operations. This includes updates by SystemUser. See the comments in the `GsBitmap class >> hiddenSetSpecifier` method; hidden sets after the one labeled "last mutable hiddenSet" generate an error if an attempt is made to modify them.

NotifySet and GCI sets

There are system hidden sets that may need to be updated by users (other than SystemUser) for particular GemStone application requirements. These hidden sets have a separate, limited public API in System class.

The #NotifySet (System hidden set 25) has an interface defined in class System with methods in the Notification category; `clearNotifySet`, `addToNotifySet:`, etc.

#ExportedDirtyObjs and #TrackedDirtyObjs (System hidden sets 22 and 23) have an interface defined in class System, category Gci Set Support. These method allow these sets to be initialized and to get and clear the contents.

#PureExportSet and #GciTrackedObjs (System hidden sets 39 and 40) also have an interface in class System, category GciSets. These methods allow you to add and remove objects from these hidden sets.

Methods returning GsBitmaps

The following methods have been added as alteratives to existing methods but return an instance of GsBitmap. There are some differences from the earlier equivalents; refer to image method comments for details.

```
Repository >> allInstances:
Repository >> fastAllInstances:
```
Analogous to `Repository >> (fast)listInstances:`

```
Repository >> allObjectsInObjectSecurityPolicies:
Repository >> fastAllObjectsInObjectSecurityPolicies:
```
Analogous to `Repository >> (fast)listObjectsInObjectSecurityPolicies:`

```
Repository >> allObjectsLargerThan:
Repository >> fastAllObjectsLargerThan:
```
Analogous to `System class >> (fast)findObjectsLargerThan:limit:`

```
Repository >> allReferences:
Repository >> fastAllReferences:
```
Analogous to `Repository >> (fast)listReferences:`

```
Repository >> allReferencesToInstancesOfClasses:
Repository >> fastAllReferencesToInstancesOfClasses:
```
Analogous to `Repository >> (fast)listReferencesToInstancesOfClasses:toDirectory:`

Changes to existing methods

The output files generated by `findDisconnectedObjectsAndWriteToFile*` methods are now in GsBitmap file format, which is different than the format used in earlier versions.

The method `loadGcCandidatesFromFile:intoHiddenSet:` requires the argument be a GsBitmap hiddenSetSpecifier, such as #CustomerSet1.

Deprecated Methods

The following methods, whose functions can be more easily performed using GsBitmap, have been deprecated:

```
Repository >> auditPageOrderOopFileWithId:
Repository >> closePageOrderOopFileWithId:
Repository >> openPageOrderOopFile:
Repository >> numberOfObjectsInPageOrderOopFileWithId:
Repository >> readObjectsFromFileWithId:startingAt:upTo:into:
```

Note that some of these are primitives and do not send the `deprecated:` message.

2.6 Other changes affecting administrative operations

fileSizeReport formatting change

The method `Repository >> fileSizeReport` returns a string describing the extents in the repository. This display has removed newlines to produce a more compressed result, and the dividing lines have been adjusted for readability.

Added method Repository >> extentsReport

This method produces a report with the same information as `fileSizeReport`, but formatted in a more condensed way with two lines per extent.

Temporary Symbol creation allowed during restore from logs

During restore, the symbol gem is not running so it is not possible to create persistent symbols. This creates problems for executing code that contains temporary variables that do not already exist as a symbol.

In v3.4, while in restore mode, you may now perform operations that create symbols, and these symbols exist only in temporary memory. If the session has created temporary symbols and does an abort, the session will be terminated with error 4070.

This only applies in restore mode: symbol creation is still not allowed in normal mode when the symbol gem is not running.

Logging to GCI server rather than client

A number of maintenance methods log output to the application console. While most of these logged to the server console, a few methods wrote to the client console, which was irretrievably lost in some configurations.

The following methods that previously logged to the client console via `GsFile >> gciLogClient:` now invoke `gciLogServer:`.

```
Repository >> reclaimAll
Repository >> _setGcConfigAt:put:
Repository >> _setupContinuousRestore:
System class >> clusterAllSymbols
```

The login log includes the Kerberos principal

When STN_LOGIN_LOG_ENABLED is set, the login log entry that is written now includes another field within each line. The new field, at the end of each line, lists the name of the KerberosPrincipal used for single sign-on login enclosed in single quotes, or '' (two single quotes) if single sign-on login was not used.

descriptionOfSession: results includes Kerberos principal

The result of `System descriptionOfSession:` now includes an additional element (22): The KerberosPrincipal object used for single sign-on login to the session, or nil if single sign-on login was not used.

Manually send LostOT to a session

The method `System class >> sendLostOtToSession:` has been added.

2.7 Cache Warming Changes

With large caches, running cache warming on startup is important to avoid unpredictable initial performance as object table and data pages are loaded into an empty cache. In v3.4, there is an improved system for specifying cache warming on startup, and a way to load the specific working set of useful data pages.

Running cache warming automatically on startstone

Historically there have been two options for cache warming: configuration parameters that specified cache warming to automatically run on startup, and a script that was normally run immediately after startup, but required manual execution.

In v3.4, configuration parameters have been added to allow the startcachewarmer script to be automatically invoked on stone startup or on remote cache startup. These new parameters let you specify the particular arguments to be used to invoke the startcachewarmer script.

The previously existing cache warming configuration parameters (STN_CACHE_WARMER and STN_CACHE_WARMER_SESSIONS) are not deprecated in this release, but will be deprecated in a future release, ensuring that all cache warming uses the same code paths.

The added configuration parameters allow automatic cache warming on:

- Stone's cache, using "STN_CACHE_WARMER_ARGS" on page 57
- All remote shared page caches, using "GEM_CACHE_WARMER_ARGS" on page 56
- All mid-level caches that are started, using "GEM_CACHE_WARMER_MID_CACHE_ARGS" on page 56

To configure automatic cache warming, the relevant subset of startcachewarmer arguments are used as the setting for the configuration parameter. For example,

```
STN_CACHE_WARMER_ARGS = "-d";
```

Legal startcachewarmer arguments for this usage include **-d -D -l -n -w**, and for remote cache warmers only, **-L**.

Cache warming based on previously loaded data pages

Cache warming options include loading the object table only into the shared page cache, or loading both the object table and data pages. By loading data pages, you can (potentially) load the entire repository into the SPC, and provide the best performance for initial queries.

However, if your cache is not large enough to hold all the data pages, the cache warmer loaded as many pages as would fit into the cache, in page id order. There was previously no way to ensure that frequently accessed pages were loaded, rather than unnecessary or archival data.

In v3.4 an argument has been added to allow you to configure your system to record the pageIds of data pages in your shared page cache to a compressed disk file, and have these data pages loaded on stone startup. Using this feature, after a stone shutdown and restart, the shared page cache can contain the same set of data pages as were loaded before shutdown, which are likely to be frequently used data.

There are two parts to this feature:

- startcachewarmer now always looks for the well-known file of working set pageIds, and loads this if the file exists. If the working set file does not exist, the usual default, **-d** or **-D** argument manages the behavior, as in previous releases.
- Using the startcachewarmer **-w** *interval* argument instructs the shared page cache monitor to write the working set file every *interval* minutes. With a value of 0, it writes the file only on clean stone shutdown (including a kill that performs a clean shutdown).

The well-known working set file is

`/opt/gemstone/locks/<stoneName><hostid>workingSet.lz4`

The stone log now includes descriptive messages on what was loaded by cache warming; the gem log file for the cache warmer has also been renamed and is now written to the same directory as other stone log files.

For example, by configuring cache warming in the stone configuration file with the following:

```
STN_CACHE_WARMER_ARGS = "-w 0 -d";
```

With this setting, the first time the stone is started, all data pages are loaded into the cache in page order; on subsequent stone restarts, the working set file exists and is loaded instead.

startstone can wait for cache warming to complete

While logins are allowed during cache warming, it may be preferable to wait for cache warming to complete before making the repository available.

You may now specify for startstone to wait until cache warming is complete, before returning, using the new configuration parameter STN_CACHE_WARMER_WAIT_MODE.

This has three options:

0 - startstone never waits for cache warming to complete.

1 - startstone waits for cache warming to complete, only after a clean shutdown or a startup without tranlogs (-N); if recovery or tranlog replay was needed, startstone returns after recovery but before cache warming completes. This is the default.

2 - startstone always waits for cache warming to complete.

See STN_CACHE_WARMER_WAIT_MODE on page 57.

waitstone can wait for cache warming to complete

The waitstone utility now has an additional positional argument that allows it to wait for cache warming to complete before returning.

See “Wait for cache warming to complete” on page 60.

2.8 Configuration Parameter changes

Parsing Change

With configuration option values that are strings, it is now allowed to use either single quotes, double quotes, or no quotes.

Quotes are required for option values containing embedded spaces.

Changes in parameters

The following option has been removed:

GEM_GCI_LOG_ENABLED

The following option has been renamed:

STN_TRAN_Q_TO_RUN_Q_THRESHOLD has been renamed, and the semantics changed. It is now STN_TRANQ_TO_RUNQ_THRESHOLD.

See “STN_TRANQ_TO_RUNQ_THRESHOLD” on page 58.

The following options have been superceded:

STN_CACHE_WARMER
STN_CACHE_WARMER_SESSIONS

The added options STN_CACHE_WARMER_ARGS, GEM_CACHE_WARMER_ARGS, and GEM_CACHE_WARMER_MID_CACHE_ARGS provide automatic cache warming on startup, invoking the cache warming scripts, and are the preferred way to configure automatic cache warming.

The older way to invoke automatic cache warming on startup, using STN_CACHE_WARMER and STN_CACHE_WARMER_SESSIONS, are still available and fully supported; however, these options may be deprecated in a future release.

STN_MAX_AIO_REQUESTS default/minimum changed

The calculation for default and minimum for STN_MAX_AIO_REQUESTS has been changed to accommodate smaller systems. Previously, the default was 128; now, the default is the lesser of SHR_PAGE_CACHE_NUM_PROCS and 128, and minimums less than 100 are allowed.

STN_NUM_LOCAL_AIO_SERVERS default changed

STN_NUM_LOCAL_AIO_SERVERS previously defaulted to 1. Now, the default is one thread per extent, up to 4 total. For more than 4 AIO page server threads, you need to explicitly configure this setting.

STN_TRAN_LOG_SIZES min value changed

The minimum value was previously 3MB, is now 10 MB.

Added configuration Parameters

GEM_CACHE_WARMER_ARGS

This parameter configures cache warming on a remote cache started by a remote gem. When this is non-empty, cache warming will run automatically when the remote cache is started.

Arguments to enable cache warming are a string that may containing white space, or the arguments used to start a cache warmer. When a remote gem creates a remote shared page cache, if the value of this configuration parameter is empty, no cache warmer is started; if the value contains spaces only, a default cache warmer is started. Otherwise the value should contain valid arguments that will be used to invoke the cache warmer.

startcachewarmer arguments may be obtained using **startcachewarmer -h**. The only arguments that should be used are: **-d -D -l -L -n -w**.

This option is only used by remote gems that create a remote shared page cache. It is ignored by all other gems.

Default: "" (cache warmer will not be started)

GEM_CACHE_WARMER_MID_CACHE_ARGS

This parameter configures cache warming on a mid-level cache. When this is non-empty, cache warming will run automatically when a mid-level cache is started.

Arguments to enable cache warming are a string that may containing white space, or the arguments used to start a cache warmer. When a gem creates a mid-level shared page cache, if the value of this configuration parameter is empty, no cache warmer is started; if the value contains spaces only, a default cache warmer is started. Otherwise the value should contain valid arguments that will be used to invoke the cache warmer.

startcachewarmer arguments may be obtained using **startcachewarmer -h**. The only arguments that should be used are: **-d -D -l -L -n -w**.

This option is only used by remote gems that create a mid-level shared page cache. It is ignored by all other gems.

Default: "" (cache warmer is not started)

GEM_COMPRESS_TRANLOG_RECORDS

WIth this configuration parameter set to true, sessions will compress tranlog data records using lz4 compression before sending them to the Stone. The resulting tranlog files are significantly smaller.

Default: true

GEM_KERBEROS_KEYTAB_FILE

Path to the Kerberos key table file. The keytab file only is required when single sign-on logins to GemStone are in use. The file contains pairs of Kerberos principals and encrypted keys. In this case, the Kerberos service is the service for a GemStone repository.

Refer to the GemStone System Administration Guide to learn about creating and maintaining this file.

Default: NONE

GEM_KEYRING_DIRS

A list of directories which contain keys and certificates used for secure backup and restore.

Runtime equivalent: #GemKeyRingDirs
Default: NONE

SHR_PAGE_CACHE_LARGE_MEMORY_PAGE_SIZE_MB

Specifies the large memory page size in megabytes used when creating the shared page cache. This option is only supported on Linux, and is ignored on other platforms. A value of 0 means use the default large page size for the host. Normally the default large page size on Linux is 2 MB, but that default may be changed by the system administrator.

Valid large page sizes on Linux are 2 MB and 1024 MB. Not all systems support one or both large memory page sizes.

This setting is ignored if SHR_PAGE_CACHE_LARGE_MEMORY_PAGE_POLICY is set to 0.

Linux: Default: 2 Min: 0 Max: 1024

STN_ALLOW_NO_SESSION_INIT

If false, GciLogin ignores the flag GCI_CLIENT_DOES_SESSION_INIT.

If true, GciLogin implements the flag GCI_CLIENT_DOES_SESSION_INIT.

Runtime equivalent: #StnAllowNoSessionInit (may only be set by SystemUser)
Default: FALSE

STN_CACHE_WARMER_ARGS

This parameter configures cache warming on the Stone's cache on Stone startup. When this is non-empty, cache warming will run automatically when Stone starts.

Arguments to enable cache warming are a string that may containing white space, or the arguments used to start a cache warmer. On Stone startup, if the value of this configuration parameter is empty, no cache warmer is started; if the value contains spaces only, a default cache warmer is started. Otherwise the list should contain valid arguments that will be used to invoke the cache warmer.

The list of valid arguments may be obtained using **startcachewarmer -h**. The only arguments that should be used are: **-d -D -l -n -w**.

If this configuration option is specified with a non empty string, it overrides any settings for STN_CACHE_WARMER and STN_CACHE_WARMER_SESSIONS

Default: "" (cachewarmer is not started)

STN_CACHE_WARMER_WAIT_MODE

Determines if and when startstone waits until cache warming has finished. The following values are allowed:

0 - disabled. startstone does not wait until cache warming has finished.
1 - startstone waits until cache warming has finished but only if stone is starting after a clean shutdown, or without tranlogs (i.e., startstone with -N option). Otherwise startstone does not wait for cache warming to finish. This is the default setting.
2 - startstone always waits until cache warming has finished.

Has no effect and is ignored if cache warming is not enabled.

Default: 1 Min: 0 Max: 2

STN_COMMIT_RECORD_BM_CACHING

When true, enables caching at the commit point of page allocation information needed when disposing a commit record. This option can reduce I/Os during commit record dispose when the commit record backlog is high, or when there is a lot of page preemption occurring in the shared cache.

When enabled, the maximum commit rate is slightly lower and commit latency is slightly higher because more work is done in the commit critical region in stone.

Default: true

STN_GEM_PGSVR_CONNECT_TIMEOUT

The time in seconds that a remote gem will wait for a connection to a pgsvr on stone's machine to complete.

Runtime equivalent: #StnGemPgsvrConnectTimeout (may only be set by System-User)
Default: 20 Min: 5 Max: 3600

STN_GEM_PRIVATE_PGSVR_ENABLED

If TRUE, a remote gem will start a private pgsvr process if the attempt to connect to a multithreaded pgsvr on stone's machine fails.

Runtime equivalent: #StnGemPrivatePgsvrEnabled (may only be set by SystemUser)
Default: FALSE

STN_GROUP_COMMITS

Specifies the maximum number of commits to group into a tranlog write. The default, 1, means commits are not grouped. Grouping is performed only if another session is waiting to commit while stone is processing a session's commit.

Default: 10 Min: 1 Max: 20

STN_TRANQ_TO_RUNQ_THRESHOLD

The number of sessions in the commit queue (waiting for the commit token) that are allowed to simultaneously process unions (read old commit records) while waiting for the commit token.

For example, if this parameter is set to 2, then sessions commitQueue[1], and commitQueue[2] (if they exist) will process unions. The first session in the commit queue , commitQueue[0], will never process unions since it will receive the token when the current commit completes.

Cache Statistic: (Stone) StnTranQToRunQThreshold
Runtime equivalent: #StnTranqToRunqThreshold (may only be set by SystemUser)
Default: 2 Min: 1 Max: 20

2.9 Utility and script changes

gemnetdebug change

The default for GS_DEBUG_VMGC_VERBOSE_OUTOFMEM is now 1, so more information will be printed by default on out of memory.

In this case, if no .csv file is configured, the .csv formatted data is now also written to the stdout (topaz console or gem log).

copydbf changes

copydbf supports lz4 compression

The **-Z** (uppercase) option has been added to allow lz4 compression for the resulting output file.

The new **-z** (lower case) option compresses using gzip, and is the same as the previously existing **-C** option.

Tranlog restore can read either lz4 or gzip, and copydbf can read both gzip and lz4 compressed files

Added secure backup options

The output for copydbf **-i**/**-I** for secure backup files includes some encryption information. In addition, the new options **-K**, **-V**, **-W**, **-X**, and **-Y** are supported only with secure backups. For more information on using these options for secure backups, see "copydbf information about secure backups" on page 41.

startnetldi

Process authentication using Kerberos

The NetLDI now supports Kerberos for client authentication, as described under "Using Kerberos for NetLDI authentication" on page 37.

To enable this, the **-k** *keytab* option has been added. When this is set, the NetLDI can use Kerberos to authenticate the host user id.

Version mismatch handling

The RPC connection to a NetLDI now uses SSL in v3.4. As a result, the NetLDI cannot definitely report a version mismatch since the connection itself does not complete. Attempting to connect to the 3.4 NetLDI with an RPC client using v3.3x or older shared libraries results in a message in the netldi log:

```
accept failed, SSL mismatch: we expect SSL, peer does not
expect SSL; client may be using old version of libgci library
```

NetLDI connection table

The NetLDI no longer uses an internal table to manage concurrent requests. The number associated with requests in the debug log (**startnetldi -d**) are now monotonically increasing, rather than reflecting concurrent requests.

startcachewarmer

Cache warming a working set

Cache warming now supports warming using a set of data pages that were previously in the cache and written to the working set file on disk. See"Cache warming based on previously loaded data pages" on page 53 for details.

If the working set file exists, with the name and location:

`/opt/gemstone/locks/`*stoneNameHostid*`.workingSet.lz4`

then the startcachewarmer will load valid pages from this file. Invalid pages will be ignored. The **-d** or **-D** options apply when that file does not exist.

Using the new option **-w** *writeInterval*, the startcachewarmer instructs the shared page cache monitor to write pages to the well-known file at the given interval, as described on page 54.

waitstone

Headers simplified

Previously, the waitstone utility printed a complete header file with all build and version details. This has been simplified; now a single line is printed. For example:

```
unixprompt> waitstone gs64stone
waitstone[Info]: GemStone version '3.4.0'
  arguments: gs64stone
Network service !#server!gs64stone is ready.
```

Wait for cache warming to complete

A fourth positional argument, *waitForWarming*, has been added.

If an integer larger than 0 is specified as the fourth argument to waitstone, waitstone will block until cache warming completes; this is useful when cache warming is automatically run on startup. Note that the specific value of the *waitForWarming* is not important.

Use of this option requires that you specify both of the preceding arguments rather than relying on the defaults.

stoplogreceiver

The undocumented and unsupported stoplogreciver **-p** argument, which specified the local port only, has been removed. The existing **-P** argument, which uses either the listening port or the local port depending on how the logreceiver was started, is the correct argument to use.

Legal to use netldid and stoned directly

startnetldi and **startstone** are utilities that invoke the `netldid` and `stoned` executables, block until startup is complete, and return control to the command line.

There are cases in which it is more useful to invoke a non-blocking utility. To support this, is it now allowed to invoke the `netldid` or `stoned` executables directly.

Note that these are in `$GEMSTONE/sys`, and thus may not normally be on the path.

2.10 Topaz Changes

Printing change

There are changes in how topaz displays characters to stdout. These changes only apply to stdout; characters that are written to files such as logged by output push are written as UTF-8, as before.

- Non-printing control characters are now returned as caret notation to avoid problems with x terminals. For example, the character with codePoint 11 is displayed as ^K.
- Characters over 255 are provided encoded. For example the character with codePoint 353, š, is displayed as \u0161.

  Note that this is a change in behavior; previous versions sent UTF-8 to the terminal for Characters over 255, but this was not reliable for xterm/shells in all environments.

Option added for linked topaz that suppresses gemnetid in .topazini

The **-L** command line argument has been added to topaz.

Using this option instead of **-l** (lowercase ell) similarly invokes linked topaz, but any setting of gemnetid within a .topazini script or a script passed in using **-I** (uppercase eye) is ignored.

Since setting gemnetid causes linked topaz to login an RPC session, and input via .topazini is not echoed, this avoids the confusion when an intended linked topaz login inadvertently becomes RPC.

Setting gemnetid explicitly by entering the topaz **set gemnetid** command, including in files input via the **input** command, is not affected by the **-L** option, and will set the linked topaz environment to login RPC.

set cachename changes

The topaz **set cachename** command sets the name of the process, as displayed in VSD.

Previously, this was only applied to linked sessions; now, both linked and RPC sessions may use **set cachename**.

Previously, the cache name had the numeric session id appended to the end of the name; now, the name is used as specified.

The cache name is still limited to 31 characters. Now, it displays a warning when the name is truncated; previously it was silently truncated.

-u sets cachename

The **topaz -u** command line argument previously did not do anything, but was useful for identifying the process in the OS.

Now, the name provided by **-u** is used to set the cachename, equivalent to using the topaz **set cachename** command.

Custom arguments following POSIX end of options marker

The topaz and gem command lines now accept custom arguments, specified using by using -- following the regular options. Any text following ' -- ' on the command line are passed to the application without interpretation.

```
-- stringOfArgs
```

These argument can be accessed, for example, by invoking a System method:

```
unixprompt> topaz -l -- foo bar
<topaz startup and login headers>
topaz 1> run
System commandLineArguments printString
%
anArray( 'topaz', '-l', '--', 'foo', 'bar')
```

Added LIST options

Added option linenumbers

List allows you to print the source of a method. The new **linenumbers** option allows you to get a listing of the method source that includes line numbers. For example,

```
topaz 1> set class Set
topaz 1> list linenumbers method: asSet
  1  asSet
  2
  3  "Returns a Set with the contents of the receiver."
  4
  5  ^ self
```

Added options primitives, cprimitives

List selectors of all methods which are primitives. These commands accept an optional string token, which restricts the list to selectors that contain the token.

primitives or **prim** lists instance methods, **cprimitives** or **cprim** lists class methods.

Added argument for selectors and cselectors

list selectors and **cselectors** allow you to find instance or class method selectors within the currently set class. Now, you may also include a token after the selectors keyword, which will be used to filter the results. For example:

```
topaz 1> set class Time
topaz 1> list cselectors mill
  fromMilliseconds:
  millisecondClockValue
  millisecondsElapsedTime:
```

Added SUBHIERARCHY command

The subhierarchy command has been added, which prints a hierarchy report for all subclasses of the current class without an argument, or of the argument class.

Added RUNBLOCK command

The topaz command **runblock** was added to support internal debugging.

runblock takes two arguments on the command line, and the following lines up to the next % must be the source for a block with between 0 and 10 block variables.

The first argument is used for self in the block, and can be anything that can be specified on the command line.

The second argument must be an Array of size N, N <= 10, where N is the number of argument variables. This second argument can be specified using ** or @OOP, or named in UserGlobals.

SESSIONINIT to allow login when session initialization fails

The topaz **set** command has a new argument **sessioninit**.

This command allows you to bypass the `GsCurrentSession initialize` execution that is normally done on every login. This is important for cases such as upgrades that require method recompile, or errors in methods invoked during initialize.

To control the ability to bypass `loginHook:`, **set sessioninit** is only allowed when the new configuration parameter STN_ALLOW_NO_SESSION_INIT has been set. This can be set during runtime only by SystemUser, using the runtime parameter #StnAllowNoSessionInit. For example:

```
System stoneConfigurationAt: #StnAllowNoSessionInit put: true
```

When this is set, additional status messages are printed to topaz during login.

When STN_ALLOW_NO_SESSION_INIT is set to true, `GsCurrentSession initialize` and the `loginHook:` block are executed, but errors do not prevent login; this allows debugging the `loginHook:` initialization code.

To turn off execution entirely, in topaz, execute:

```
set sessioninit off
```

This disables execution of `GsCurrentSession initialize` for subsequent logins.

2.11 Cache Statistics Changes

statmonitor changes

New option to specify the hours for restart

The `-k` *listOfTimes* option has been added.

This sets statmonitor to restart each day at specified times. Either the **-r** or **-R** flag must also be specified, and neither **-h** nor **-t** flags may be specified. The list of times must start and end with a single quote and times must be separated by a comma. Hours must be specified in 24 hour format.

The times do not need to be in order, but should not contain duplicates. The restart will be performed at the next sampling interval after the time, so may be delayed with long sample intervals.

For example, to restart statmonitor at 1:30 am, 9:30 am and 1:30 pm:

```
statmonitor -r -k '01:30,9:30,13:30' gs64stone
```

statmonitor supports lz4 compression

The option **-Z** has been added, to write output in compressed lz4 output.

VSD v5.3 and later support reading .lz4 compressed statmonitor files.

Deprecated options removed

The deprecated options -s0, -s1, -s2, -s3, and -s4 have been removed.

The following statistics have been removed:

StnAioMainTimeInAioWrite

StnAioWritesQueuedCount

LogRecordsIoCount (replaced by **TranlogIoCount)**

LogRecordsWritten

RcRetryQueueSize (replaced by **RcTransQueueSize)**

TranlogRecordKind

TranlogRecordsWritten

Changes in existing statistics

NewGenSizeBytes, **OldGenSizeBytes**, **PermGenSizeBytes**, and **CodeCacheSizeBytes** are now 64 bit.

The following statistics have been added:

ActiveCHeapLogBuffers (Stone)
Number of C Heap tranlog buffers in stone for which tranlog writes are in progress.

CommitQueueHeadNoMsg (Stone)
The number of times a session in commit queue was not ready for token and not ready for service.

CommitQueueHeadNotReadyCount (Stone)
The number of times the session at head of commit queue was not ready to receive the commit token.

CommitQueueNotSerializing (Stone)
The number of times a session in commit queue was not in serialization.

CommitQueueSymbolWait (Stone)
The number of times a session in commit queue was waiting for SymbolGem to commit.

CommitRecordsDisposedNotCached (Stone)
The number of commit records not in stone's CR cache at the time they were disposed.

FreeLogBuffers (Stone)
Number of free tranlog buffers in shared memory.

MemMappedSize (Gem)
The size in bytes of memory mapped regions used by multithreaded operations

NumWorkingSetWrites (SPC Monitor)
Number of writes to the cache warmer working set file performed since the system was started.

RcTransQueueSize (Stone)
Shows the number of sessions waiting for an rc commit to complete so that their rc retry can be serviced.

StnAioCompletedHeapBuffers (Stone)
Number of tranlog buffers written from C heap memory.

StnAioCompletedSharedBuffers (Stone)
Number of tranlog buffers written from shared memory.

StnAiosWaitedForWriteThread (Stone)
Number of times a tranlog write was delayed because all tranlog write threads were busy.

TranlogBuffersWritten (Gem, Stone)
The number of tranlog buffers written to the tranlogs.

TranlogIoCount (Stone)
The number of pwritev calls made by stone to write to the transaction logs since stone was last started.

AIX System stats

On AIX, added System Pages statistics

The following stats are now available, in a new category "System Pages". They are only collected if -A or -W is specified.

These SystemPages stats are collected using vmgetinfo(), vs. AIX_System stats which are collected using the perfstat API.

AvailableMemory
Number of bytes of memory available without paging. From the memavailable member of the vminfo64 struct returned by the vmgetinfo() AIX function.

ClientSegPages
Count of pages in use for the client segment. From the numclseguse member of the vminfo64 struct returned by the vmgetinfo() AIX function.

ClientSegPagesPinned
Count of pages pinned for the client segment. From the numclsegpin member of the vminfo64 struct returned by the vmgetinfo() AIX function.

LoanedPages
Number of pages loaded to the hypervisor for CMO. From the nloaned member of the vminfo64 struct returned by the vmgetinfo() AIX function.

MemGuardRemovedFail
Number of pages mguard failed to remove. From the memgrd_fail_pgs member of the vminfo64 struct returned by the vmgetinfo() AIX function.

MemGuardRemovedOk
Number of pages mguard successfully removed. From the memgrd_succ_pgs member of the vminfo64 struct returned by the vmgetinfo() AIX function.

NonRemovablePages
Number of nonremovable pages for DR. From the normlmbmem member of the vminfo64 struct returned by the vmgetinfo() AIX function.

NonSystemPages
Number of pages on SCBs not marked V_SYSTEM. From the nonsys_pgs member of the vminfo64 struct returned by the vmgetinfo() AIX function.

PageAheadMax
Maximum number of page ahead pages. From the maxpgahead member of the vminfo64 struct returned by the vmgetinfo() AIX function.

PageAheadMin
Minimum number of page ahead pages. From the minpgahead member of the vminfo64 struct returned by the vmgetinfo() AIX function.

PageOutsFileBuferRemote
Count of file buffer remote page outs. From the numremote member of the vminfo64 struct returned by the vmgetinfo() AIX function.

PageOutsFileBuffer
Count of file buffer page outs. From the numpout member of the vminfo64 struct returned by the vmgetinfo() AIX function.

PageSpaceFreeBlocks
Number of free paging space blocks. From the psfreeblks member of the vminfo64 struct returned by the vmgetinfo() AIX function.

PagesRepagedComp
Number of computational pages repaged. From the nrepaged member of the vminfo64 struct returned by the vmgetinfo() AIX function.

PagesRepagedFile
Number of file buffer pages repaged. From the nrepaged member of the vminfo64 struct returned by the vmgetinfo() AIX function.

PagesReplacedComp
Number of computational pages replaced. From the nreplaced member of the vminfo64 struct returned by the vmgetinfo() AIX function.

PagesReplacedFile
Number of file buffer pages replaced. From the nreplaced member of the vminfo64 struct returned by the vmgetinfo() AIX function.

PersistSegPages
Count of pages in use for the persistent segment. From the numpseguse member of the vminfo64 struct returned by the vmgetinfo() AIX function.

PersistSegPagesPinned
Count of pages pinned for the persistent segment. From the numpsegpin member of the vminfo64 struct returned by the vmgetinfo() AIX function.

PinnablePages
Number of pages available for pinning. From the pfavail member of the vminfo64 struct returned by the vmgetinfo() AIX function.

PinnablePagesAppLevel
Number of pages available for pinning by applications. From the pfpinavail member of the vminfo64 struct returned by the vmgetinfo() AIX function.

RemoteAllocations
Number of remote allocations. From the numralloc member of the vminfo64 struct returned by the vmgetinfo() AIX function.

RepagedCompFrames
Count of repaged computational frames. From the rpgcnt member of the vminfo64 struct returned by the vmgetinfo() AIX function.

RepagedFileFrames
Count of repaged files frames. From the rpgcnt member of the vminfo64 struct returned by the vmgetinfo() AIX function.

SegmentIdHw
Max index + 1 of sids ever used. From the hisid member of the vminfo64 struct returned by the vmgetinfo() AIX function.

SigDangerThreshold
Threshold of free memory pages below which the operating system may send SIGDANGER to processes. From the npswarn member of the vminfo64 struct returned by the vmgetinfo() AIX function.

SigKillThreshold
Threshold of free memory pages below which the operating system may send SIGKILL to processes. From the npskill member of the vminfo64 struct returned by the vmgetinfo() AIX function.

SoftPagesClientSeg
Number of soft pages in use for the working segment. From the soft_clseguse member of the vminfo64 struct returned by the vmgetinfo() AIX function.

SoftPagesPersistSeg
Number of soft pages in use for the working segment. From the soft_pseguse member of the vminfo64 struct returned by the vmgetinfo() AIX function.

SoftPagesWorkingSeg
Number of soft pages in use for the working segment. From the soft_wseguse member of the vminfo64 struct returned by the vmgetinfo() AIX function.

SpecialDataSegIds
Number of special dataseg id's. From the numspecsegs member of the vminfo64 struct returned by the vmgetinfo() AIX function.

SpecialDataSegIdsFree
Number of free special dataseg id's. From the numspecfree member of the vminfo64 struct returned by the vmgetinfo() AIX function.

SpecialDataSegIdsHighWater
Highwater count of special dataseg id's. From the specsegshi member of the vminfo64 struct returned by the vmgetinfo() AIX function.

SystemPages
Number of pages on SCBs marked V_SYSTEM. From the system_pgs member of the vminfo64 struct returned by the vmgetinfo() AIX function.

SystemReservedBlocks
Number of system reserved blocks. From the pfrsvdblks member of the vminfo64 struct returned by the vmgetinfo() AIX function.

TrueFreePages
True number of free 4K memory pages. From the true_numfrb member of the vminfo64 struct returned by the vmgetinfo() AIX function.

TrueMemPages
True total number of logical and physical 4K memory pages. From the true_memsizepgs member of the vminfo64 struct returned by the vmgetinfo() AIX function.

UnmanagedPages
Number of pages not on SCBs. From the unmngd_pgs member of the vminfo64 struct returned by the vmgetinfo() AIX function.

WorkingSegPages
Count of pages in use for the working segment. From the numwseguse member of the vminfo64 struct returned by the vmgetinfo() AIX function.

WorkingSegPagesPinned
Count of pages pinned for the working segment. From the numwsegpin member of the vminfo64 struct returned by the vmgetinfo() AIX function.

The following are additional AIX_System stats:

Backtracks
Count of backtracks. From the backtrks member of the vminfo64 struct returned by the vmgetinfo() AIX function.

ExecFilledPages
Count of exec filled pages. From the exfills member of the vminfo64 struct returned by the vmgetinfo() AIX function.

ExtendXptWaits
Count of extend XPT waits. From the extendwts member of the vminfo64 struct returned by the vmgetinfo() AIX function.

FreeFrameWaits
Count of free frame waits. From the freewts member of the vminfo64 struct returned by the vmgetinfo() AIX function.

FreeListSize
Number of pages in the free list. From the numfrb member of the vminfo64 struct returned by the vmgetinfo() AIX function.

LargePagesFree
Number of free large memory pages. From the lgpg_numfrb member of the vminfo64 struct returned by the vmgetinfo() AIX function.

LargePageSize
Size in bytes of a large memory page on this host. From the lgpg_size member of the vminfo64 struct returned by the vmgetinfo() AIX function.

LargePagesTotal
Total number of large memory pages. From the lgpg_cnt member of the vminfo64 struct returned by the vmgetinfo() AIX function.

LargePagesUsed
Number of large memory pages in use. From the lgpg_inuse member of the vminfo64 struct returned by the vmgetinfo() AIX function.

LargePageUsedHighWater
High water number of large memory pages used. From the lgpg_hi member of the vminfo64 struct returned by the vmgetinfo() AIX function.

LockMisses
Count of lock misses. From the lockexct member of the vminfo64 struct returned by the vmgetinfo() AIX function.

MaxClient
The maximum number of pages that may be used for client pages. From the maxclient member of the vminfo64 struct returned by the vmgetinfo() AIX function.

MaxFree
The threshold of free pages above which the page-stealing algorithm will stop stealing pages to replenish the free list. From the maxfree member of the vminfo64 struct returned by the vmgetinfo() AIX function.

MaxPerm
The threshold of free pages below which the page-stealing algorithm may steal

computational pages as well as file buffer pages. If the free page list is larger than this value, then the page-stealing algorithm will only steal file buffer pages. From the maxperm member of the vminfo64 struct returned by the vmgetinfo() AIX function.

MinFree
The threshold of free pages below which the page-stealing algorithm will begin to steal pages to replenish the free list. From the minfree member of the vminfo64 struct returned by the vmgetinfo() AIX function.

MinPerm
The threshold of free pages below which the page-stealing algorithm may steal computational and file buffer pages regardless of the repaging rate. From the minperm member of the vminfo64 struct returned by the vmgetinfo() AIX function.

NumClient
Number of client frames. From the numclient member of the vminfo64 struct returned by the vmgetinfo() AIX function.

NumPerm
Number of pages used to cache files. From the numperm member of the vminfo64 struct returned by the vmgetinfo() AIX function.

PageReclaims
The total number of page reclaims; both from the free list and from disk. Page reclaims are caused by a reference to a page that has been stolen from a process by the page daemon.

PendingIoWaits
Count of pending I/O waits. From the pendiowts member of the vminfo64 struct returned by the vmgetinfo() AIX function.

RealMemoryPages
The real memory size in 4K pages. From the memsizepgs member of the vminfo64 struct returned by the vmgetinfo() AIX function.

VirtualPagesAccessed
The number of virtual pages accessed. From the numvpages member of the vminfo64 struct returned by the vmgetinfo() AIX function.

ZeroFilledPages
The total number of pages have been block-cleared to contain all zeros.

Chapter

3 GemStone Smalltalk changes and new features

This chapter describes changes and new features important for programmers using GemStone Smalltalk, including:

3.1 Changes in Class and Method handling

Changes to class variables requiring class versioning

Starting in v3.1, changing class variables did not require creating a new version of the class in the case of a class with only a single version. However, if multiple versions of the class existed, a new version was required when class variables were changed.

This restriction for multiple-versioned classes has been lifted.

Added Behavior method

The following method has been added:

`Behavior >> classVarAt:` *aClassVar* `otherwise:` *defaultValue*

Changes in class history for String and Symbol classes

In previous releases, the class histories for the various String and Symbol classes included the single and multiple byte variants, e.g., String's classHistory included both DoubleByteString and MultiByteString. This was designed for the historical constraints, migration, and support for `isKindOf:` and similar operations.

As of 3.4, class histories conform to the new rule:

> At most one entry in any class history shall have (Globals at: entry name) == entry

The various String classes now have a classHistory size of 1.

`isKindOf:` and similar methods have been reimplemented to avoid any change in behavior.

changeClassTo: handling of differently sized objects

The method `Object >> changeClassTo:` can be used to change the class of an object, including between classes with different numbers of named or indexed instance variables (indexed or varying instance variables are collection elements).

Previously, there was some undesirable behaviors in the cases when the number of and types of variables did not match:

- Previously, when the number of named instVars in the target class was greater than the number of named instance variables in the receiver instance, nils were inserted for the additional named instVars, between the slots for the named and indexed values. Now, indexed values are moved into the named slots.
- Indexed instVars remained on the receiver including cases in which the target class did not support indexed instance variables. Now, this is disallowed.
- If the number of named instVars was fewer than the number of named instance variables in the target, then extra named instance variable values moved to indexed slots. Now, this is disallowed.

The previous behavior is available using `Object >> legacyChangeClassTo:`.

`changeClassTo:` now behaves according to the following rules:

- If the instance has more total instvars (combined named and indexed) than the number of named instVars defined by the target class, then the target class must support indexed instVars (i.e. a collection). The overflow named or indexed instance variables will move into indexed instance variable slots.
- If the instance has more total instvars than the number of named instVars defined by the target class, and the target class does not support indexed instance variables, it is an error.
- If the target class defines more named instVars than the number of named instVars in the instance, and the instance has indexed instVars, then the data in the indexed instVars is moved into the otherwise empty named instVars slots. Any indexed instance variables beyond those required to fill the named instVars of the target class will remain indexed instance variables, with their offsets reduced by the number of indexed instance variables that became named instance variables
- If the target class defines more named instVars than the total named and indexed instVars in the instance, then the extra slots are initialized to nil.

GsNMethod printOn:

GsNMethod now implements `printOn:`. As a result, displaying a GsNMethod now includes the class and selector of the method.

Pragmas

A pragma is a literal selector or keyword message pattern that occurs between angle brackets at the start of a method after any temporaries. For example:

```
<foo: 123 >
<foo: 5 bar: 'update'>
<bar>
```

Pragma keywords follow method selector syntax, but they are symbol literals within the method, not message sends. Pragmas are useful to provide metadata about methods and to support keyword searching.

While primitive directives in GemStone look like pragmas, they are not; and `primitive:` is a reserved word in the first pragma in a method.

Some pragma support was introduced for seaside-based environments in v2.x, but the implementation was incomplete and not usable in the base GemStone environment. (#43541).

Pragma class added

The Pragma class provides a way to find out information about pragmas. An instance of Pragma references the method that defines it, and the keyword and argument or arguments.

Sending `GsNMethod >> pragmas` will return an array of instances of Pragma.

Pragma class methods provide search capabilities. `Pragma >> allNamed:in:` returns a collection of all Pragmas with the given keyword in methods in the given class.

For other search methods, see the image.

Added ClassOrganizer subhierarchy report

The following method has been added:

```
ClassOrganizer >> subhierarchyReport:includeOops:
```

For example:

```
topaz 1> run
ClassOrganizer new subhierarchyReport:  PositionableStream
      includeOops: true
%
PositionableStream   19773697 ( collection position readLimit)
  ReadStream   19774721
  WriteStream   19775745 ( writeLimit)
    CypressMessageDigestStream   20292865
    ReadWriteStream   19776769
      FileStream   19777793 ( gsfile streamType)
```

Support for multiple execution environments

GemStone includes the ability to differentiate multiple execution environments, primarily with the intention to allow support for Ruby. These are distinguished using the `envId:` argument. GemStone Smalltalk execution uses environment 0.

In v3.4, some methods have been added to increase operability in environments 1 and above, although this is not fully supported for general use. Image methods with ruby-specific names have been removed.

Behavior/Module

The following methods have been added as variants of existing methods:

```
allSuperClassesForEnv:
compileMethod:dictionaries:category:intoMethodDict:
   intoCategories:environmentId:
superclassForEnv:
superclassForEnv:put:
```

(only for use with environments > 0)

ClassOrganizer

ClassOrganizer now can be used for lookups in environments other then 0, using the class instance creation methods:

```
newForEnvironment:
newWithRoot:forEnvironment:
newWithRoot:forUserId:forEnvironment:
```

PrivateObject

This is a new class, a subclass of nil (i.e., it does not inherit from Object). This is an abstract superclass for internal, hidden classes such as LargeObjectNode and NscNode.

Instance of subclasses of PrivateObject are normally for internal use only, but may be returned from some methods, such as reference path code. Only messages implemented in PrivateObject can be sent to instance of subclass of PrivateObject; other messages will signal a MessageNotUnderstood.

3.2 Changes in Collections

species, speciesForCollect, speciesForSelect

While usually `collect:`, `select:`, `reject:`, and similar methods return instances of the same kind of collection as the receiver, in some cases this is not correct. These overrides are normally handled by class-specific implementations of `species`, `speciesForCollect`, and `speciesForSelect`. In v3.4, the implementations have been cleaned up to move behavior into `species*` methods and remove unnecessary duplicates.

In addition, some RC classes incorrectly returned a result in an RC collection, rather than the equivalent non-RC collection, based on the return from `speciesForCollect`. (#46870)

Added RC Collection classes

The classes RcArray, RcIdentitySet, GsPipe, and RcPipe have been added.

RcIdentityBag has been reimplemented to be more efficient, with the name RcLowMaintenanceIdentityBag; both classes are available.

RcArray

The class RcArray is similar to Array, but no conflict occurs when multiple users add objects to an RcArray using specific methods. If a conflict with other update operations on the RcArray occur, the add is replayed so that the commit can succeed.

Only the following methods support concurrent updates

`add:`
`addAll:`
`at:put:` (where no other session affects the element at the at: index)
`size:` (when size is increased)

RcLowMaintenanceIdentityBag

RcLowMaintenanceIdentityBag is a reimplementation of RcIdentityBag that avoids the use of per-session add and remove sets. This avoids the periodic manual maintenance that is required in some cases for RcIdentityBag. Like RcIdentityBag, there is no commit conflict with multiple sessions adding and a single session removing objects.

While the API is similar, the performance characteristics will vary between RcIdentityBag and RcLowMaintenanceIdentityBag. As for v3.4, both classes exist. In a future release, it is expected that RcIdentityBag will be replaced by RcLowMaintenanceIdentityBag, and the current implementation be renamed.

Due to the implementation, while you can create indexes on an instance of RcLowMaintenanceIdentityBag, in this release only indexes of type btreePlusIndex are supported, not legacyIndexes. See “Indexes on new Reduced-conflict classes” on page 84.

RcIdentitySet

For completeness with the other reduced-conflict classes, RcIdentitySet has been added. This provides an interface similar to Set, but using replay to handle commit conflicts. There is no commit conflict with multiple sessions adding and a single session removing objects.

Due to the implementation, while you can create indexes on an instance of RcIdentitySet, in this release only indexes of type btreePlusIndex are supported, not legacyIndexes. See “Indexes on new Reduced-conflict classes” on page 84.

GsPipe

The class GsPipe implements a first-in-first-out queue, with no conflict when a single session adds objects to the GsPipe, and only one session removes objects.

Internally, the GsPipe is implemented as a linked list of GsPipeElements. So, while most RC classes provide the reduced-conflict via either replaying conflicting transactions, or via per-session subcollections, GsPipe inherently has no conflict between operations at either end of the list.

RcPipe

The class RcPipe implements a first-in-first-out queue, with no conflict when multiple sessions add objects to the RcPipe, and only one session removes objects.

Internally, the RcPipe is implemented as a linked list of GsPipeElements. Unlike with GsPipe, if a conflict with an add by another session occurs, the add operation is replayed so that the commit can succeed. Only add: and operations that invoke add: are reduced-conflict.

Improved Concurrency for RC replay

Reduce-conflict classes use replay to resolve conflicts. Previously, all RC replay was serialized using a single lock, referenced by the global variable GemStoneRCLock.

Now, the Stone manages serialization so that only RC replays that conflict are serialized.

The global GemStoneRCLock has been removed, and the method `System class >> waitForRcWriteLock:`, which was intended for internal use in RC locking, has been removed.

The cache statistic **RcRetryQueueSize** is replaced by **RcTransQueueSize**.

Return type for remove: and add:

ANSI specifies that `add:` and `remove:` should return the argument, not the receiver; there were a small number of GemStone collection classes that did not respect this.

ANSI does not specify a return type for `addAll:` or `removeAll:`; most GemStone collection classes return the receiver for these cases.

The following methods have been updated in v3.4 to return the argument, not the receiver.

```
ExceptionSet >> add:
RcIdentityBag >> remove:
RcIdentityBag >> remove:ifAbsent:
Bag >> removeAll:
RcIdentityBag >> removeAll:
Set >> removeAll:
```

You should review your application code to ensure that any uses of the return value from these methods are updated.

Set arithmetic operators extended to Set and Bag

Set and Bag now implement intersection (*), union (+), and difference (-) operations. Previously these were only available in IdentityBag and its subclasses.

Added SequencableCollection method

The following instance method has been added to SequencableCollection:

`indexOfSubCollection:` *aSubColl*
: Returns the index of the first element of the receiver where that element and the subsequent ones are equal to those in *aSubColl*. The search is begun in the receiver starting at 1.

Added Append classes for performance

The following classes provide optimized behavior for using Streams to assemble text.

AppendStream

AppendStream is a kind of Stream that is only for append, that is, performing writes at the end, and later retrieving the entire contents by sending `contents`. It is not "positionable", and implements only a limited set of Stream protocol. This allows it to be significantly faster for some common use cases, such as using a stream to compose complex text.

AppendableString

AppendableString is a kind of String that understands AppendStream protocol as well as String protocol. This allows use of Stream protocol without redirection through a stream, and may provide performance benefits for some uses.

While non-stream methods are inherited from String and not disallowed, they are not intended to be used.

This class may be subject to change.

Legacy stream added methods

The PortableStream hierarchy contains useful methods that were not previously available in the LegacyStream classes. The following methods have been added:

```
WriteStreamLegacy >> crlf
WriteStreamLegacy >> crtab
WriteStreamLegacy >> crtab:.
WriteStreamLegacy >> space:
WriteStreamLegacy >> tab:
ReadStreamLegacy >> nextMatchFor:
```

Added String/UTF8 optimization and encoding

The following methods have been added for performance:

`String >> addAllUtf8:` *aCharacterOrString*
`Utf8 >> addAllUtf8:` *aCharacterOrString*
Append the UTF8 encoding of the argument to the receiver.

`CharacterCollection >> addCodePoint:` *aSmallInteger*
`DoubleByteString >> addCodePoint:` *aSmallInteger*
`String >> addCodePoint:` *aSmallInteger*
`Unicode16 >> addCodePoint:` *aSmallInteger*
`Unicode7 >> addCodePoint:` *aSmallInteger*
Append the character with the given codePoint to the receiver.
This is implemented a primitive for String, DoubleByteString, Unicode 7 and Unicode 16.

encodeAsUTF8IntoString

The following methods have been added:

```
String >> encodeAsUTF8IntoString
DoubleByteString >> encodeAsUTF8IntoString
QuadByteString >> encodeAsUTF8IntoString
```

Previously there were private methods `_encodeAsUTF8intoString`, with the same behavior (note the capitalization changein `i`, as well as the underscore). These underscore versions remain for compatibility.

javaScriptEncode

Strings can now be encoded in Javascript, that is, with control characters escaped, by using the added method `String>>` and `MultiByteString>>javaScriptEncode`.

For details on the behavior of these methods, see the class methods `String >> _javascriptCharacters` and `_javascriptEncode:`.

Symbol parsing

Previously, expressions such as these:

```
#(a + b)
#(a - 3b)
```

interpreted the binary operator and variables in various ways, or errored.

Per ANSI, this is illegal syntax, and are now disallowed. Now, only legal identifiers or keywords can omit the leading #. Legal binary selector symbols must include #, and creating other symbols requires both # and quotes.

For example, assuming the intention is to create three-element arrays, to create the equivalents of the above expressions now requires:

```
#(a #+ b)
#(a #- #'3b')
```

While this example is in a literal Array, the change involves all cases of symbol parsing.

Note that with this change, GemStone still does not conform entirely to ANSI, which would require a leading # for keywords (which have a final colon). Expressions such as `#(a: 3)` retain the same, non-ANSI behavior.

keywords

Symbol keywords changes

`Symbol >> keywords` parses the receiver by $:, returning an Array of the selector keywords with each substring including the trailing $:. Previously, this had a number of shortcomings.

- If any of the terms (sequences of characters delimited by colons) was not a legal selector keyword, previously this method returned an empty array. Now, the results may include strings that are not legal selectors, such as keywords beginning with a digit or including characters such as $-.

- Multiple colons were handled as if they were one colon. Now, each colon is treated as finalizing a term.
- Previously, for a receiver with no colon (a unary selector), `keywords` appended a colon. (bug #46552)

CharacterCollection isKeywords more precise

`CharacterCollection >> isKeyword` continues to use a more precise ANSI definition of keyword, and only returns true for legal keyword selectors, not for unary or binary selectors. This also now detects some cases of illegal secondary keywords that previously did not return false. (#46696)

3.3 Indexing changes

The code that supports querying and indexing has been intensively reviewed and some parts have been reimplemented, and extensive testing has been added. Many bugs have been fixed, particularly in corner cases and in enumerated and set-valued indexes.

Terminology and parallel features

In v3.2, GemStone added a new indexing API the **GsIndex/GsQuery API**, to allow expansion on the **historic API** that uses UnorderedCollection methods and selectBlocks. Both APIs produce the same results, but many new features are not accessible from the historic API.

With v3.4, GemStone adds a new indexing internal structure, the **BtreePlus inde**x, based on a Btree+ implementation, to improve on the **Legacy index** internal structures based on simple Btrees and the RcIndexDictionary. Both internal implementations are available and (of course) produce exactly the same query results, but have different performance profiles and differences in default and error behavior.

Your application may use both the GsIndex/GsQuery API and the traditional API, and include both BtreePlus and Legacy internal structures. However, you cannot use both BtreePlus and Legacy indexing structures when creating multiple indexes on the same collection.

BtreePlus indexes

The data structures that supported indexing in previous releases included two data structures: Btrees, which contain the mappings for equality and allow range queries, and an RcIndexDictionary, which was used for mappings for identity queries and to map from the last element in the Btree node to the object itself.

v3.4 includes a new implementation of indexing data structures, the BtreePlus index. This btree+ based internal data structure includes the mappings that were previously done in the RcIndexDictionary, so no dictionary is needed.

The legacy indexing structures that were provided in previous releases are still available, and are the default for upgraded applications.

Using a BtreePlus indexes or a legacy index provides the same query results, but there are some differences in performance profiles and in default and error behavior. BtreePlus indexes perform best in combination with the new option optimizedComparisons, but there are more restrictions on the specific mixes of data types when optimizedComparisons are specified.

Optimized comparisons

Optimized comparisons is a new feature that is key to achieving the performance benefits of BtreePlus indexes.

Historically, equality indexes may have lastElementClass objects that are of different classes but which can be ordered against each other, such as Strings and Symbols, Floats and NaNs, or any object and nil. However, while less-than and greater-than queries are not a problem, these objects may never be equal; for example, both ('aa' <= #aa) and ('aa' >= #aa) are true, but ('aa' = #aa) is not true. To support these cases, extra message sends are done for equality indexes (the issue, obviously, does not arise for identity indexes).

To avoid the extra comparison costs when basic comparisons (e.g. =) are sufficient, you may now specify the index option #optimizedComparisons.

optimizedComparisons only apply to btreePlusIndexes, not legacyIndexes.

To use the optimizedComparison option, the values must be a kind of the last element class. Nil is **not** allowed as a last element class value, and the following rules apply:

- For Float last element class, NaNs are not allowed as a value.
- For String last element class, Symbols are not allowed as a value. Unicode strings are only allowed if the repository is in Unicode Comparison Mode.
- For Symbol last element class, Strings are not allowed as a value.

When specifying optimizedComparison for the index, you will get an error if an element in the collection does not follow these rules; for example, if any element in the collection has a nil in the instance variable that the index specifies. If your collection legally may include nils in indexed instance variables, or if you mix Strings and Symbols, you cannot use the optimizedComparison option, and using legacy Indexes is likely to provide better performance.

Note that nils along the path, or missing instance variables as allowed by the existing GsIndexOptions optionalPathTerms setting, are unaffected by optimizedComparison. optimizedComparison specifies what is in the btreePlus holding the last element class values, while nils along the path affect other lookup structures.

lastElementClass

With legacy indexes, a lastElementClass of CharacterCollection, String, or Symbol (or the multi-byte equivalents) created indexes that allowed values of any of the above to be used as an indexed value. Specifying a lastElementClass of Unicode7, 16, or 32 created a unicode index with a collator; and in Unicode Comparison Mode, Unicode strings could be mixed with Traditional strings and symbols as indexed values. This behavior is the same for legacy indexes in v3.4.

However, using btreePlusIndex and optimizedComparison, mixing strings and symbols is disallowed. When using optimizedComparison, the meanings of the lastElementClass specification does distinguish between the "flavor" of the string.

When optimizedComparison is specified (either explicitly, or set as default), then:

- Specifying a lastElementClass of String, DoubleByteString, or QuadByteString creates an optimizedComparison index that disallows symbols. If the repository is not in Unicode Comparison Mode, Unicode strings are also disallowed.
- Specifying a lastElementClass of Symbol, DoubleByteSymbol, or QuadByteSymbol creates an optimizedComparison index that disallows strings.
- Specifying a lastElementClass of Unicode7, Unicode16, or Unicode32 creates an optimizedComparison index that disallows symbols. If the repository is not in Unicode Comparison Mode, Traditional strings are also disallowed.
- Specifying a lastElementClass of CharacterCollection creates a **legacyIndex** , not an optimizedComparison index. Since you cannot mix classes in an optimizedComparison index, the performance will be better with legacyIndex than with btreePlusIndex without optimizedComparison. However, note that you cannot mix legacy indexes and btreePlus indexes on a given collection.

  The use of CharacterCollection is not recommended.

The following added convenience methods create an btreePlusIndex with optimizedComparison, regardless of any GsIndexOptions defaults, and restrict the lastElementClass values to the specified "flavor" of String:

```
stringOptimizedIndex:
stringOptimizedIndex:options:
symbolOptimizedIndex:
symbolOptimizedIndex:options:
symbolOptimizedIndex:collator:
symbolOptimizedIndex:collator:options:
unicodeStringOptimizedIndex:
unicodeStringOptimizedIndex:options:
unicodeStringOptimizedIndex:collator:
unicodeStringOptimizedIndex:collator:options:
```

Performance

btreePlusIndex and optimizedComparison are designed to significantly improve the performance of indexed queries, at the expense of somewhat less performant index updates. The performance improvements from these new indexed will depend on the performance profile of your application.

If the performance of indexed queries in your application is satisfactory, there is no need to change; the performance profile of legacyIndex is the same in v3.4, and overall performance is likely to be improved by other changes in v3.4.

If you are having issues with query performance, we recommend experimenting with the btreePlusIndexes and optimizedComparison, and verifying the performance benefits and ensuring that update performance does not create separate problems.

Note that btreePlusIndex without optimizedComparison is likely to be less performant than legacyIndexes.

GsIndexOptions changes

The instance of GsIndexOptions associated with every index has additional importance in this release, since it controls the use of btreePlusIndex or legacyIndex. One or the other of these options must always be set. In addition, optimizedComparison can only apply when btreePlusIndex is set, and is strongly recommended to get the performance improvements of the btreePlusIndex.

The way the defaults for GsIndexOptions are handled has been changed to provide more levels of control between the three basic types (legacyIndex, btreePlusIndex with optimizedComparison, and btreePlusIndex without optimizedComparison). The defaults also support the other options available in previous releases.

Given the multiple levels of default, the way instances are combined has been updated to support intuitively mathematical operators of +, -, and the new `not` operator.

Internally, GsIndexOptions has fields for each option that may be nil, true, or false. When instances are combined, the option fields that are true or false in the second operand take precedence.

When using default, the order of precedence is:

- GsIndexOptions passed in with the `options:` keyword
- default set using `GsIndexOptions sessionDefault:`
- default set using `GsIndexOptions default:`. This is set to legacyIndex by upgrade.
- "background" default of btreePlusIndex with optimizedComparison

not operator

The not operator has been introduced to improve clarity in specifying options.

For example, if the default is:

(GsIndexOptions btreePlusIndex + GsIndexOptions optimizedComparison)

you can turn off optimized compare using:

(GsIndexOptions optimizedComparison not)

Auto-unset of optimizedComparison

Since optimizedComparison is only valid for btreePlusIndexes, when a GsIndexOptions is set to legacyIndex (or btreePlusIndex not), then optimizedComparison is automatically turned off.

Initial default

The initial default for upgraded applications is:

(GsIndexOptions legacyIndex)

The initial default for new repositories is:

(GsIndexOptions btreePlusIndex + GsIndexOptions optimizedComparison)

Each instance of GsIndexOption is based on this default. So, for example, for a new repository, creating an instance of reducedConflict using the expression:

(GsIndexOptions reducedConflict)

the resulting GsIndexOptions instance would be:

(GsIndexOptions btreePlusIndex + GsIndexOptions optimizedComparison + GsIndexOptions reducedConflict

To avoid the default, you may explicitly specify the state for each option. This is easily done using the not operator.

Setting the default

You can change the initial default by executing either of the following:

`GsIndexOptions default:` *aGsIndexOptions*
: Persistently change the default GsIndexOptions for all users. This method can only be executed by SystemUser.

`GsIndexOptions sessionDefault:` *aGsIndexOptions*
: Set the default GsIndexOptions for this session only.

For example, if your initial default is:

(GsIndexOptions btreePlusIndex + GsIndexOptions optimizedComparison)

setting a session default to:

(GsIndexOptions optimizedComparison not + GsIndexOptions reducedConflict)

will result in GsIndexOptions instances being created by default as:

(GsIndexOptions btreePlusIndex + GsIndexOptions reducedConflict)

The btreePlusIndex stays the same, the optimizedComparison is removed, and the reducedConflict is added.

Indexes on new Reduced-conflict classes

This release introduced several new Reduced-Conflict (RC) classes, including two UnorderedCollection subclasses, RcIdentitySet and RcLowMaintenanceIdentityBag. In this release, you can create btreePlusIndexes on these classes, but you may not create legacyIndexes. This is due to the way conflict resolution is implemented in the new classes.

Queries on Array and other non-UnorderedCollections

It is now allowed to use instances of GsQuery to search over collections that do not support indexes, such as Array. Performance will be comparable to iterative search, but it allows managing queries similarly over various collection types and using the GsQuery syntax to flexibly build queries.

Most, but not all, of the collection classes may be queried using GsQuery. Class methods have been added to indicate if GsQueries are allowed:

```
_canCreateQueryOnInstances
```

Improved reliability and bug fixes

While the new substructure was developed and new features implemented, extensive additional testing has been added, and a large number of issues have been fixed; particularly relating to streamed queries, enumerated and set-value queries and queries involving nil values. These are not individually described here.

3.4 Socket Enhancements and new features

GsSocket adds support for SO_REUSEPORT

The SO_REUSEPORT socket option permits multiple AF_INET or AF_INET6 sockets to be bound to an identical socket address. This option may not be available on all OS revisions.

GsSecureSocket

GsSignalingSocket

The class GsSignalingSocket has been added, a subclass of GsSocket. This class signals kinds of Error, rather than returning nil, when a socket operation fails.

Errors now signalled, rather than methods returning false

GsSecureSocket is now a subclass of GsSignalingSocket.

Now when using GsSecureSocket, when an error occurs, the methods signal a SocketError or SecureSocketError rather than returning false.

This is true for most but not all error conditions; in particular, methods that return true or false, such as `GsSecureSocket >> readWillNotBlock`, will return nil on error.

GsSocket behavior is unchanged.

SSLv3

SSLv3 connections are no longer supported, as they are known to be insecure; only TLSv1 connections are currently supported.

Additional Peer Authentication Options

GsSecureSocket server sockets now support the verification modes SSL_VERIFY_FAIL_IF_NO_PEER_CERT and SSL_VERIFY_CLIENT_ONCE, which can be used when server certificate verification is enabled.

New methods in the image allow you to set options to an array that may include:

- #SSL_VERIFY_FAIL_IF_NO_PEER_CERT -- if the client did not return a certificate, the TLS/SSL handshake is immediately terminated with a 'handshake failure' alert.
- #SSL_VERIFY_CLIENT_ONCE -- only request a client certificate on the initial TLS/SSL handshake. Do not ask for a client certificate again in case of a renegotiation.

These options are only supported for server sockets; there are no verification options for client sockets.

The following methods have been added

`GsSecureSocket class >> fetchCertificateVerificationOptionsForServer`
Answers an Array of Symbols which represent the certificate verification options that can be used by server sockets.

`GsSecureSocket class >> setCertificateVerificationOptionsForServer:` *anArray*
Sets the certificate verification options for server sockets using an Array of Symbols. If *anArray* is empty then any previously set options are cleared. Certificate verification for server sockets must be enabled before executing this method. Only applies for sockets created after this method is executed.

`GsSecureSocket >> fetchCertificateVerificationOptions`
Answers an Array of Symbols which represent the certificate verification options that can be used by the receiver. Certificate verification options are only supported by server sockets.

`GsSecureSocket >> setCertificateVerificationOptions:` *anArray*
Sets the certificate verification options for the receiver, which must be a server socket, using an Array of Symbols. If *anArray* is empty then any previously set options are cleared. The receiver must enable certificate verification before this

method is executed, and must use this method before the receiver attempts a connection with its peer.

The following are added protocol to check for the current status of verification:

`GsSecureSocket class >> certificateVerificationEnabledOnClient`
Answer true if certificate verification is enabled for client sockets.

`GsSecureSocket class >> certificateVerificationEnabledOnServer`
Answer true if certificate verification is enabled for server sockets.

`GsSecureSocket >> certificateVerificationEnabled`
Answer true if certificate verification is enabled on the receiver.

Methods have also been added to specify a certificate directory.

`GsSecureSocket class >> useCACertificateDirectoryForClients:` *aDirectoryString*
Specifies a directory where trusted certificate authority (CA) certificates in PEM format are located. The certificates in this directory will be used to authenticate certificates provided by servers during the SSL handshake. The directory may contain more than one certificate.

Certificates are loaded into the internal SSL state which is valid for the current session only; the SSL state is not committed to the repository. Has no effect on instances created before the method was called, nor on server sockets.

If successful, this method also enables certificate verification for client sockets as if the #enableCertificateVerificationOnClient method was called.

`GsSecureSocket class >> useCACertificateDirectoryForServers:` *aDirectoryString*
Specifies a directory where trusted certificate authority (CA) certificates in PEM format are located. The certificates in this directory will be used to authenticate certificates provided by client during the SSL handshake. The directory may contain more than one certificate.

Certificates are loaded into the internal SSL state which is valid for the current session only; the SSL state is not committed to the repository. Has no effect on instances created before the method was called, nor on client sockets.

If successful, this method also enables certificate verification for server sockets as if the #enableCertificateVerificationOnServer method was called.

Client certificate verification automatically enabled when CA certificate set

`GsSecureSocket >> useCACertificateFileForClients:` and `useCACertificateFileForServers:` now enable cerificateVerification; previously this required an additional step for clients.

SecureSocketError

The class SecureSocketError has been added, and is used for security errors signaled from GsSecureSocket.

SocketError asString

SocketError asString has been added, providing a printed version that is specific to SocketError.

HTTPS example

A https client example has been added, demonstrating a client SSL socket connection to https://google.com with full certificate verification enabled.

Connect with timeout

The following methods have been added, to allow a timeout to be specified on connect.

```
GsSecureSocket >> secureAcceptTimeoutMs:
GsSecureSocket >> secureConnectTimeoutMs:
```

3.5 Numerics changes

Random default change

The Random class is an abstract class, with actual random numbers generated by subclasses HostRandom, Lag1MwcRandom, and Lag25000CmwcRandom.

In previous releases, Random class methods `new` and `seed:` returned instances of Lag25000CmwcRandom. This implementation of random number generator took significant time to create initially, and was designed to be used where a series of reliably random numbers was needed. The large overhead of creation, however, made it unsuitable for cases in which a single random number was required, and a truly random number was not required. (#46990)

This has been changed in v3.4. Now, `Random new` will return an instance of HostRandom, and `Random seed:` returns an instance of Lag1MwcRandom.

For applications that have strict requirements for randomness, it is recommended to refer to the specific class directly, e.g. `Lag25000CmwcRandom seed:` .

FloatingPoint signalling Exceptions

For compliance with the IEEE 754 spec on floating point exceptions, starting with GemStone v3.0, certain floating point operations that signalled exceptions in v2.x instead returned an exception float, such as a NaN or Infinity. For example, dividing by 0.0 returned PlusInfinity rather than erroring. This can create potentially serious problems for applications that rely on exceptions to detect errors. (#46398)

Now, you may configure GemStone such that floating point operations that by default return the IEEE 754-specified ExceptionalFloat will instead signal an Exception.

The following are the symbols representing the kinds of errors for which exception signalling can be enabled:

- #divideByZero
- #overflow
- #underflow
- #invalidOperation
- #inexactResult

To configure the set of errors for which Exceptions are signalled, use class protocol for FloatingPointError.

`enableAllExceptions`
Enable exceptions for all kinds of errors

`enabledExceptions`
Return an Array containing zero or more of the error symbols, reflecting the most recent call to #enableAllExceptions or #enableExceptions:

`enableExceptions:` *symOrArrayOfSym*
The argument may be one of the error symbols, or an Array containing zero or more of these Symbols , or nil. After an arithmetic operation when an exception occurs, it will check and if the corresponding symbol is enabled, it will signal a FloatingPointError, otherwise it will return the ExceptionalFloat.

Use care when enabling all exceptions, in particular enabling #inexactResults, as this may impact your application unexpectedly. When #inexactResults is enabled, expressions such as `1.0 / 3` will error.

Some non-floating point operations also use FloatingPointError, such as LargeInteger overflow. These exceptions are signaled regardless of FloatingPointError status, since there is no appropriate kind of number to return.

Added methods

In addition to this way of controlling exceptions, several other methods have been added:

`Number, Float >> roundedNoFpe`
Returns the Integer nearest in value to the receiver, ignoring any #inexactResult that might be enabled in FloatingPointError

`Number, Float >> truncatedNoFpe`
Returns the integer that is closest to the receiver, on the same side of the receiver as zero is located. ignoring any #inexactResult that might be enabled in FloatingPointError

`Float >> noInexactResultDo:` *aBlock*
Return value of executing aBlock, suppressing any FloatingPointError for #inexactResult that might occur.

```
Number >> isExceptionalFloat

Float >> isExceptionalFloat

DecimalFloat >> isExceptionalFloat
```

Invariance for non-special numeric instances

For numbers that are not specials, such as LargeIntegers, Fractions, and ScaledDecimals, some code paths could return an instance that was not invariant. (#46790). This does not apply to specials/immediate: SmallInteger, SmallDouble, and SmallFraction.

Now, all numbers are return as invariant.

#postCopy methods have been added to the non-special classes.

Obsolete methods

The following methods on BinaryFloat class have been replaced by the new FloatingPointException handling. While these methods are technically only deprecated in this release, they are no longer functional; you should update any references to these methods in your code as part of the upgrade to 3.4.

`clearAllExceptions`	`operationExceptions`	`roundingMode:`
`clearException:`	`raisedException:`	`status`
`enabledExceptions`	`raisedExceptions`	`status:`
`on:do:`	`roundingMode`	`trapEnabled:`
`operationException:`		

3.6 GsEventLog

GsEventLog is a shared, reduced-conflict logging tool that allows multiple sessions to:

- record errors and informational messages to a single location
- flexibly include objects as well as text
- easily and flexibly query and sort by particular attributes.

A GsEventLog class variable holds a instance of GsEventLog, which in turn holds a collection of log entries, which are instances of kinds of GsEventLogEntry. It is allowed for user applications to create custom subclasses of GsEventLogEntry.

Each event has a priority. Built in priorities are fatal, error, warning, info, debug, and trace; these are stored internally as integers.

The entries are stored in an instance of RcArray, allowing concurrent writes of log entries. Note that the order of elements is based on the order in which the commits occurred, while entry timestamps reflect the time at which the entry was created.

Adding events

GsEventLog may hold both application (user) events and system events. User entries can be added in two ways: class convenience methods such as `logError:`, `logInfo:`, etc., or by creating an instance of GsEventLogEntry and sending `addToLog`. A commit is required to make the log entry persistent.

System events should be added only by GemStone code. In this release, GemStone code does not write System events.

Querying and reporting

To create a string containing text representation of the entire contents, send

```
GsEventLog current report
```

To search for a subset of entries with particular attributes

```
(GsEventLog current entriesSatisfying: aBlock) report
```

Deleting events

GsEventLog is not reduced conflict for delete. It is recommended to lock the log using `GsEventLog >> acquireGsEventLogLock`. The lock is cleared automatically on commit.

You can clear all events using the following method:

```
GsEventLog current deleteAllEntries.
```

By querying for specific methods, you can delete those methods using `deleteEntry:` or `deleteEntries:`

It is possible to restrict modifying or removing events. To do this, execute

```
GsEventLog entriesUnmodifiable
```

After this is executed, new entries to the log are made invariant and the standard delete methods will not delete them. However, they are not protected from delete using private delete protocol.

System events are also protected from modification or deletion, other than using private delete protocol.

Example

```
topaz 1> run
[GsEventLog logDebug: 'About to perform divide by zero'.
 1 / 0.
 GsEventLog logDebug: 'After divide by zero'.
 true]
   on: Error
   do: [:ex | GsEventLog logObject: ex.  ex resume].
%
true
topaz 1> run
GsEventLog current report
%
'2017-07-13 16:36:44.3820  24198 Trace  About to perform divide by zero
2017-07-13 16:36:44.3821  24198 Trace  a ZeroDivide occurred (error
  2026), reason:numErrIntDivisionByZero, attempt to divide 1 by zero
  a ZeroDivide occurred (error 2026), reason:numErrIntDivisionByZero,
  attempt to divide 1 by zero
2017-07-13 16:36:44.3822  24198 Trace  After divide by zero
'
```

3.7 Changes in Errors and Error Handling

Message class added

The class Message has been added, for improved ANSI compatibility in handling does not understand. Previously, two-element arrays held the class and selector.

Now, `doesNotUnderstand:` is invoked with an instance of Message, and sending `#message` to MessageNotUnderstood error returns an instance of Message.

For compatibility, Message understands `at:`, `size`, and `asArray`, so existing code that expects a two-element array will continue to work correctly.

#rtErrInvalidArgClass now maps to ArgumentTypeError

Previously error 2283, invoked as #rtErrInvalidArgClass, mapped to ArgumentError; now this maps to ArgumentTypeError.

"statement has no effect" compiler errors

Previously code that included statements with no effect triggered a compiler warning and so did not compile. (#42510)

Now, a warning is triggered in this case for method compilation. Do-its may ignore the warning.

Unicode at:put: argument errors

The errors returned for invalid input to Unicode7 and Unicode16 `at:put:` arguments has changed. Previously these provided an OutOfRange error 2723 with an unclear message; now, they signal a ArgumentError 2004, or OffsetError 2003.

New errors

2518 #authErrStoreTrav /AUTH_ERR_STORE_TRAV
Smalltalk class: SecurityError
Error reported on an attempt to change the object security policy via store traversal.

2608 rtErrGroupAlreadyExists / RT_ERR_GROUP_ALREADY_EXISTS
Smalltalk class: ImproperOperation
Attempt to create a group when the group already exists.

2609 rtErrPrincipalAlreadyExists / RT_ERR_PRINCIPAL_ALREADY_EXISTS
Smalltalk class: ImproperOperation
Attempt to create a kerberos principal when the principal already exists

2755 rtErrGsSecureSocket / ERR_SecureSocketError
Smalltalk class: SecureSocketError (new in v3.4)
Certain security-related errors on a GsSecureSocket.

4016 #gsErrSslShutdown / GS_ERR_SSL_SHUTDOWN_ERR
Smalltalk class: SecureSocketError (new in v3.4)
Network error when switching from SSL to non SSL during login

4030 #errLostOtGci / ERR_LOST_OT_GCI
Smalltalk class: ImproperOperation
LostOt while waiting for a GCI command, and the transactionMode not transactionless

Changes in errors

2080 rtErrCantChangeClass, Illegal attempt to change the class of an object.
This error now includes a third argument: the reason why the change was illegal .

Removed errors

The following errors are no longer defined:

2050 repErrReplicateOnline

2037 classErrConstraintNotClass

2062 rtErrCannotChgConstraint (still referenced in image)

2107 objErrConstraintViolation (still referenced in image)

2132 classErrBadConstraint

2156 repErrReplicateNotMounted

2148 rtErrInvalidConstraintForMigration (was previously only referenced from image, now entirely removed)

2159 rtErrInvalidElementConstraintForMigration

2223 rtErrStrToPathConstraint
2308 rtErrBagNoConstraintAlongPath
2320 rtErrBadConstraintForMigration
2327 rtErrLastConstraintNotBoolean
2340 objErrDictConstraintViolation (still referenced in image)

Debugger Changes

Debugger frame printing of activations in blocks is simplified

Previously, the class of the block was included, which cluttered the view.

GsProcess added methods

For support for GCI application debugging, the following methods have been added:

```
GsProcess >> gciStepIntoFromLevel:
GsProcess >> gciStepOverFromLevel:
GsProcess >> gciStepThruFromLevel:
```

These replace the deprecated `GCIStep()` and `GciStep_()` GCI functions.

GsProcess stack report line length

The `lineLimit:` keyword has been added to the most general form of `stackReportToLimit*`:

```
GsProcess >> stackReportToLevel:withArgsAndTemps:andMethods:
   includeSource:lineLimit:
```

Other `stackReportToLevel*` methods now use a default line size of 100.

Also, for the `stackReportToLevel*` methods that have the keyword `includeSource:`, this accepts an additional argument, #block, allowing display of the source for Executed Blocks.

3.8 ProfMonitor changes

Default interval changed

Previously, the default interval was 1 millisecond (defaultIntervalNs returned 1000000). Now, it is 0.5 milliseconds (i.e. 500 microseconds); defaultIntervalNs returns 500000.

ProfMonitor reporting change

The way ProfMonitor and ProfMonitorTree generate reports has been changed to allow more control over the specific reports. New methods have been added to support this; existing methods return reports as in previous versions.

A number of private reporting methods have been removed, and the flow of control through class methods has been cleaned up.

Report headers now include the reporting threshold.

Report Generation added methods

The following methods have been added:

```
ProfMonitor class >> monitorBlock:downTo:intervalNs:reports:
ProfMonitor class >> monitorBlock:reports:
ProfMonitor >> reportDownTo:reports:
```

These methods accept the additional keyword reports:, which allows you to specify an array of report symbols. These symbols are:

`#samples` - sample counts report is included in the resulting output.
`#stackSamples` - stack sampling report is included in the resulting output.
`#senders` - method senders report is included in the resulting output.
`#objCreation` - Enables object creation tracking; the object creation report is included in the resulting output.
`#tree` - causes ProfMonitorTree to be used for profiling, and the method execution tree report is included in the resulting output.
`#objCreationTree`- causes ProfMonitorTree to be used for profiling, and enabled object creation tracking; the object creation tree report is included in the resulting output.

ProfMonitor save and report

It is now supported to save an instance of ProfMonitor and perform the analysis and report later.

In order to perform the analysis later, it is necessary that the underlying raw results file not be deleted. The convenience `monitorBlock*` methods that return reports delete the raw data file; so to allow later analysis, use the new method `ProfMonitor >> runBlock:`.

For example,

```
run
UserGlobals at: #aProfMon put:
  (ProfMonitor runBlock: [
    200 timesRepeat: [System myUserProfile dictionaryNames]]).
%
commit
logout
login
run
aProfMon reportAfterRun
%
```

Renamed instance variable

The ProfMonitor instance variable rawSampleArray has been renamed to numSamples.

spyOn: removed

The deprecated method `spyOn:` has been removed.

3.9 Other Additions and Changes

Public objectForOop:

The private method `_objectForOop:` has been in use as a way to retrieve objects based on OOP. While "oop fishing" has risks, it is a useful tool for users who understand the issues and risks.

A public version, `Object >> objectForOop:`, has been added. This method differs from `_objectForOop:` in that it errors if no object with the given OOP exists. This avoids the ambiguity when resolving using nil's OOP.

DateAndTime with ScaledDecimal seconds

By default, DateAndTime now returns an instance with the seconds as a SmallDouble, allowing arbitrary resolution (to system limits). However, this produced more digits of resolution than was needed.

The following method has been added:

`DateAndTime class >> nowWithScale:` *anInteger*
Creates a DateAndTime for the current time, using a ScaledDecimal with the given scale as the subsecond resolution. For example, using a scale of 3 will produce millisecond resolution.

To allow application-wide configuration of DateAndTime scale with a particular scale, the following new method can be used:

`DateAndTime class >> setDefaultScale:` *anInteger*
If this is executed as SystemUser with *anInteger* 0<= n <= 30000, this value will be used by subsequent DateAndTime >> now method invocations to create a DateAndTime with that given ScaledDecimal scale. If this was not set, or is set to nil, the behavior is unchanged, and DateAndTime >> now returns an instance with SmallDouble seconds.

performOnServer: and GsHostProcess optimization

These operations now use `vfork()` instead of `fork()`, and are much faster; one test demonstrated a 40x performance improvement.

GsBitmap added

The class GsBitmap has been added, encapsulating behavior of GemStone's hidden sets. While GsBitmap can be considered as a collection and implements Collection protocol, it does not inherit from Collection. Instances of GsBitmap cannot be committed, and you cannot add temporary objects nor specials to a GsBitmap.

GsBitmaps are most useful when doing repository analysis. See "GsBitmap" on page 49 for more information on using GsBitmaps.

ExecBlock

ExecBlock >> on:do: accepts zero argument do: block

The `do:` argument handling block passed to an `on:do:` expression may now be a zero argument block, as well as the usual one argument block.

Added cull:* methods

The methods `cull:`, `cull:cull:`, `cull:cull:cull:`, and `cull:cull:cull:cull:` have been added, for compatibility with Pharo and VW.

These are similar to `value:`, `value:value:`, etc, but allow for blocks that accept fewer arguments than the number of `cull:` arguments.

TestSuite debug

The following methods have been added:

```
TestSuite >> debug
TestCase >> debug
TestCase >> run
```

`TestCase >> printString` now produces the debug invocation of that case, as with GsTestCase.

Cypress subsystem preview added

Cypress is a dialect-independent solution for managing Smalltalk source code. FileTree is the GemStone version of this project.

A number of Cypress classes have been added. These are preliminary; future releases may make extensive changes to make internal use of these for code management.

Chapter

4 Changes in the GCI Interfaces

4.1 Changes in standard GCI API

Added functions

GciUtf8To8bit

```
(BoolType) GciUtf8To8bit(
    const char* src,
    char *dest,
    ssize_t destSize
    );
```

This function converts Utf8 input in *src to 8 bit data in *dest. If all code points in *src are in the range 0..255, and the result fits in destSize-1, returns TRUE and *dest is null terminated, otherwise returns FALSE.

Function changes

GciPerformFetchBytes returns a result of type ssize_t rather thant int64

A number of new reserved OOPS have been added for classes that are new in v3.4.

The GciObjRepHdrSType field objectSecurityPolicyId must now be UNDEFINED_ObjectSecurityPolicyId, if the object already exists.

Removed functions

The following deprecated functions have been removed, since they functionally require constraints, and constraint implementation is now removed.

`GciPathToStr`

`GciStrToPath`

4.2 FFI-related changes

Create a CCallout from a function pointer

The following method has been added. This allows a function pointer stored into a CByteArray by C code to be used to synthesize a CCallout from that pointer.

`CByteArray >> ccalloutAt:` *zeroBasedOffset* `name:` *aString* `result:` *resType* args: *argumentTypes*

Returns an instance of CCallout. *aString* is used to fill in the fName in the result, but not otherwise used. The 8 byte pointer value in the receiver at *zeroBasedOffset* is used instead of the result of a dlsym() call when building the result. *resType* and *argumentTypes* must be as documented in `CCallout class >> library:name:result:args:`.

CByteArray pointerAt:resultClass: allows CPointer

This method previously only allowed the `resultClass:` argument to be a CByteArray or a subclass of CByteArray. Now, CPointer may be specified and this method will create an instance of CPointer.

Copying CByteArray

Executing `CByteArray>>copyFrom:from:to:into:` with an end index that is less than the startindex returned an error. For consistency with String `copyFrom:to:`, this is now allowed.

This method returns the receiver. For consistency with String copy methods that return the number of bytes copied, the following method has been added:

`CByteArray >> copyBytesFrom:from:to:into`

4.3 Bug Fixes

GciRtlLoad with null path failed on Windows and for 32-bit GCI applications

When the path argument to GciRtlLoad is NULL, it uses a default path. This previously used the 64-bit UNIX path, `$GEMSTONE/lib`. On Windows, the shared libraries are distributed in the `bin` directory, and for 32-bit applications, the library path is `$GEMSTONE/bin32`. In these cases, GciRtlLoad did not find the libraries and failed. Now, the correct path is computed on Windows. (#46506)

GciStoreTravDoTravRefs error if same oop in both GCed set and noLongerReplicated set

If the same oop is both in arguments for the set of OOPs GCed and the noLongerReplicatedSet, it results in an error rtErrNotInExportSet. (#46883)

It was possible to encounter this if a client GBS process was terminated at the wrong moment.

4.4 Changes in Thread-safe GCI

Status of User Actions and GsFile operations

The thread-safe GCI does not support client-side user actions. This means that thread-safe GCI applications also cannot evaluate Smalltalk code that performs client-side GsFile operations.

New Thread-Safe feature to Fork execution

The following API calls have been added to support forked execution:

GciTsForkContinueWith

```
(BoolType) GciTsForkContinueWith(
    GciSession sess,
    OopType gsProcess,
    OopType replaceTopOfStack,
    GciErrSType *continueWithError,
    int flags,
    GciErrSType *err,
    GciTsCallbackFType *callback
    );
```

Execute GciTsContinueWith in a separate C thread, calling *callback when finished. Function result is TRUE if call initiated, FALSE if call could not be started, with details in *err.

GciTsForkExecute

```
(BoolType) GciTsForkExecute(
    GciSession sess,
    const char sourceStr,
    OopType sourceOop,
    OopType contextObject,
    OopType symbolList,
    int flags,
    ushort environmentId / normally zero ,
    GciErrSType *err,
    GciTsCallbackFType *callback
    );
```

Execute GciTsExecute in a separate C thread, calling *callback when finished. Function result is TRUE if call initiated, FALSE if call could not be started, with details in *err.

GciTsForkLogin

```
(BoolType) GciTsForkLogin(
    const char *StoneNameNrs,
    const char *HostUserId,
    const char *HostPassword,
    BoolType hostPwIsEncrypted,
    const char *GemServiceNrs,
    const char *gemstoneUsername,
    const char *gemstonePassword,
    unsigned int loginFlags,
    int haltOnErrNum,
    GciErrSType *err,
    GciTsCallbackFType *callback
    );
```

Execute GciTsLogin in a separate C thread, calling *callback when finished. Function result is TRUE if call initiated, FALSE if call could not be started, with details in *err. Upon completion, the first argument to the callback is the newly allocated session and second argument is a SmallInteger value 0 for clean login, 1 if a warning was returned in *err

GciTsForkPerform

```
(BoolType) GciTsForkPerform(
    GciSession sess,
    OopType receiver,
    OopType aSymbol,
    const char selectorStr,
    const OopType *args,
    int numArgs,
    int flags,
    ushort environmentId / normally zero ,
    GciErrSType *err,
    GciTsCallbackFType *callback
    );
```

Execute GciTsPerform in a separate C thread, calling *callback when finished. Function result is TRUE if call initiated, FALSE if call could not be started, with details in *err.

GciTsForkStoreTravDoTravRefs

```
(BoolType) GciTsForkStoreTravDoTravRefs(
    GciSession sess,
    const OopType *oopsNoLongerReplicated,
    int numNotReplicated,
    const OopType *oopsGcedOnClient,
    int numGced,
    GciStoreTravDoArgsSType *stdArgs,
    GciClampedTravArgsSType *ctArgs,
    GciErrSType *err,
    GciTsCallbackFType *callback
    );
```

Execute GciTsStoreTravDoTravRefs in a separate C thread , calling *callback when finished. Function result is TRUE if call initiated, FALSE if call could not be started, with details in *err. Upon completion, the second argument to the callback is the int result of GciTsStoreTravDoTravRefs expressed as a SmallInteger.

Other Added Thread-Safe functions

GciTsCallInProgress

```
(int) GciTsCallInProgress(
    GciSession sess,
    GciErrSType *err
    );
```

Returns 1 if a call is in progress on the specified session, 0 if a call is not in progress, -1 if sess is invalid in which case *err contains the details.

GciTsFetchTraversal

```
(int) GciTsFetchTraversal(
    GciSession sess,
    const OopType *theOops,
    int numOops,
    GciClampedTravArgsSType *ctArgs,
    GciTravBufType *travBuff,
    int flag,
    GciErrSType *err
    );
```

Performs a traversal starting at the oops specified by theOops and numOops, as specified by *ctArgs and flags, returning object reports in *travBuff . Returns 1 if traversal completed, 0 if data returned but traversal not complete, -1 if error returned in *err (in which case *travBuff undefined). If result == 1, call GciTsMoreTraversal again to fetch the remainder of the traversal result.

4.5 Updated Compile and Link Information

Compile and link changes in this release are minor.

Linux Compile and Link Information

Complier version

Red Hat 6.x: gcc/g++ 4.4.7

Red Hat 7.x: gcc/g++ 4.8.5

Ubuntu Linux 14.04: gcc/g++ 4.8.4

Ubuntu Linux 16.04: gcc/g++ 5.4.0

SUSE Linux 12: gcc/g++ 4.8.5

Debugger version

Red Hat 6.x: GNU gdb 7.2-90.el6

Red Hat 7.1: GNU gdb 7.6.1-80.el7

Ubuntu Linux 14.04: GNU gdb 7.7.1

Ubuntu Linux 16.04: GNU gdb 7.11.1

SUSE Linux 12: GNU gdb 7.9.1

Compiling a user action or GCI application

```
g++ -fmessage-length=0 -fcheck-new -O3 -ggdb -m64 -pipe
   -D_REENTRANT -D_GNU_SOURCE -pthread -fPIC -fno-strict-aliasing
   -fno-exceptions -I$GEMSTONE/include -x c++ -c userCode.c
   -o userCode.o
```

The following warn flags are recommended for compilation:

```
   -Wformat -Wtrigraphs -Wcomment -Wsystem-headers -Wtrigraphs
   -Wno-aggregate-return -Wswitch -Wshadow -Wunused-value
   -Wunused-variable -Wunused-label -Wno-unused-function
   -Wchar-subscripts -Wmissing-braces -Wmissing-declarations
   -Wmultichar -Wparentheses -Wsign-compare -Wsign-promo
   -Wwrite-strings -Wreturn-type -Wuninitialized
```

Linking a user action library

```
g++ -shared -Wl,-Bdynamic,-hliburAct.so userCode.o
   $GEMSTONE/lib/gciualib.o -o liburAct.so -m64
   -Wl,--no-as-needed -lpthread -Wl,--as-needed
   -lcrypt -ldl -lc -lm -lrt -lpam -lpam_misc -Wl,-z,muldefs
   -Wl,--warn-unresolved-symbols
```

Linking a GCI application

```
g++ userCode.o $GEMSTONE/lib/gcirtlobj.o -Wl,-traditional
   -Wl,--warn-unresolved-symbols -m64 -Wl,--no-as-needed -lpthread
   -Wl,--as-needed -lcrypt -ldl -lc -lm -lrt -lpam -lpam_misc
   -Wl,-z,muldefs -o userAppl
```

Solaris on x86 Compile and Link Information

Complier version

CC: Studio 12.5 Sun C++ 5.14 SunOS_i386 2016/05/31

Debugger version

Sun DBX Debugger 7.7 SunOS_i386 2009/06/03

Compiling a user action or GCI application

```
CC -xO4 -m64 -xarch=generic -Kpic -mt -D_REENTRANT
   -D_POSIX_PTHREAD_SEMANTICS -I$GEMSTONE/include
   -features=no%except -c userCode.c -o userCode.o
```

Linking a user action library

```
CC -m64 -xarch=generic -G -Bsymbolic -h libuserAct.so -i userCode.o
   $GEMSTONE/lib/gciualib.o -o libuserAct.so -Bdynamic -lc
   -lpthread -ldl -lrt -lsocket -lnsl -lm -lpam -lCrun -z nodefs
```

Linking a GCI application

```
CC -xildoff -m64 -xarch=generic -i userCode.o
   $GEMSTONE/lib/gcirtlobj.o -z nodefs -Bdynamic -lc -lpthread
   -ldl -lrt -lsocket -lnsl -lm -lpam -lCrun -o userAppl
```

AIX Compile and Link Information

Complier version

AIX 6.1, and 7.1 on POWER7: IBM XL C/C++ for AIX, V11.1

AIX 7.1 on POWER8: IBM XL C/C++ for AIX, V13.1.2

Debugger version

dbx

Compiling a user action or GCI application

```
xlC_r -O3 -qstrict -qalias=noansi -q64 -+ -qpic
   -qthreaded -qarch=pwr6 -qtune=balanced -D_LARGEFILE64_SOURCE
   -DFLG_AIX_VERSION=version -D_REENTRANT -D_THREAD_SAFE
   -qminimaltoc -qlist=offset -qmaxmem=-1 -qsuppress=1500-010:1500
   -029:1540-1103:1540-2907:1540-0804:1540-1281:1540-1090 -qnoeh
   -I$GEMSTONE/include -c userCode.c -o userCode.o
```

Depending on your version of AIX, you need to include either -DFLG_AIX_VERSION=61 or -DFLG_AIX_VERSION=71.

Also note that there is no space in the `-qsuppress` arguments that are continued on the following line.

Linking a user action library

```
xlC_r -G -Wl,-bdatapsize:64K -Wl,-btextpsize:64K
   -Wl,-bstackpsize:64K -q64 userCode.o $GEMSTONE/lib/gciualib.o
   -o libuserAct.so -e GciUserActionLibraryMain -L/usr/vacpp/lib
   -lpthreads -lc_r -lC_r -lm -ldl -lbsd -lpam -Wl,-berok
```

Linking a GCI application

```
xlC_r -Wl,-bdatapsize:64K -Wl,-btextpsize:64K
   -Wl,-bstackpsize:64K -q64 userCode.o $GEMSTONE/lib/gcirtlobj.o
   -Wl,-berok -L/usr/vacpp/lib -lpthreads -lc_r -lC_r -lm -ldl
   -lbsd -lpam -Wl,-brtllib -o userAppl
```

DARWIN Compile and Link Information

Complier version

Apple LLVM version 6.0 (clang-600.0.56)

Debugger version

GNU gdb 6.3.50.20050815-cvs (Thu Nov 21 15:33:19 UTC 2013)

Compiling a user action or GCI application

```
g++ -fmessage-length=0 -O3 -ggdb -m64 -pipe -fPIC
   -fno-strict-aliasing  -D_LARGEFILE64_SOURCE -D_XOPEN_SOURCE
   -D_REENTRANT -D_GNU_SOURCE -I$GEMSTONE/include -x c++
   -c userCode.c -o userCode.o
```

The following warn flags are recommended for compilation:

```
   -Wformat -Wtrigraphs -Wcomment -Wsystem-headers -Wtrigraphs
   -Wno-aggregate-return -Wswitch -Wshadow -Wunused-value
   -Wunused-variable -Wunused-label -Wno-unused-function
   -Wchar-subscripts -Wconversion -Wmissing-braces -Wmultichar
   -Wparentheses -Wsign-compare -Wsign-promo -Wwrite-strings
   -Wreturn-type
```

Linking a user action library

```
g++ -dynamiclib userCode.o $GEMSTONE/lib/gciualib.o
   -o libuserAct.dylib -m64 -lpthread -ldl -lc -lm -lpam -undefined
   dynamic_lookup
```

Linking a GCI application

```
g++ userCode.o $GEMSTONE/lib/gcirtlobj.o -undefined dynamic_lookup
   -m64 -lpthread -ldl -lc -lm -lpam -o userAppl
```

Windows Compile and Link Information

Complier/Debugger version

Microsoft Visual Studio 2010 Version 10.0.30319.1 RTMRel

Microsoft Visual C++ 2010 01021-532-2002102-70611

Compiling a GCI application

```
cl /W3 /Zi /MD /O2 /Oy- -DNDEBUG /TP /nologo /D_LP64 /D_AMD64_
   /D_CONSOLE /D_DLL /DWIN32_LEAN_AND_MEAN
   /D_CRT_SECURE_NO_WARNINGS /EHsc
   /DNATIVE /I 'VisualStudioInstallPath\atlmfc\include'
   /I 'VisualStudioInstallPath\VC\include'
   /I 'C:\Program Files (x86)\Microsoft
   SDKs\Windows\v7.0A\Include'
   /I '%GEMSTONE%\include' -c userCode.c -FouserCode.obj
```

Linking a GCI application

```
link /LIBPATH:"VisualStudioInstallPath\VC\lib\amd64"
   /LIBPATH:"C:\Program Files(x86)\Microsoft
   SDKs\Windows\v7.0A\Lib\x64" -RELEASE
   /OPT:REF /INCREMENTAL:NO /MAP /nologo /MANIFEST
   /MANIFESTFILE:userAppl.exe.manifest
   /MANIFESTUAC:"level='asInvoker'" userCode.obj
   %GEMSTONE%\lib\gcirpc.lib ws2_32.lib netapi32.lib advapi32.lib
   comdlg32.lib user32.lib gdi32.lib kernel32.lib winspool.lib
   Secur32.lib /out:userAppl.exe
```

Chapter

5 Bug Fixes

The following bugs in v3.3.6 are fixed in v3.4.

Idle Gems not terminated by STN_GEM_TIMEOUT

With the configuration parameter STN_GEM_TIMEOUT configured to a non-zero value, Idle Gems should be terminated once the timeout has elapsed; this was not occurring. (#46723)

Idle gems in transactionless mode could be terminated by sigAbort

A gem in transactionless mode automatically responds to sigAbort by aborting. However, if this gem was entirely idle, it did not have a chance to perform this processing, and it could be killed by lostOT handling. (#46290).

AIO page server errors on fsync may cause Stone to hang

When the AIO page servers encounter an error during fsync of the extents, such as running out of disk space, the stone could hang. fsync is a critical operation; now, the stone will shut down under this circumstance.

Additional details on I/O errors are also now printed to the stone log. (#46734, #46735)

Out of tranlog disk space during checkpoint may cause Stone to hang and restart to fail

When the out of disk space error occurs during a checkpoint, it was possible for the stone to hang. In this case, the final tranlog was not completely written and could not be read on stone restart. (#46727)

Tranlog record containing only one selectiveAbort skipped by restore

When a tranlog record contains only one record, a single selective abort, the size of the record is small enough that it is skipped by restore. (#46695)

Extra ShrPcMon processes on heavily loaded system

On a heavily loaded system, in which swapping is taking place, there a race condition in acquiring an exclusive lock on the .LCK file. This can allow multiple ShrPcMonitor processes to be started; the Stone assumes that the earlier one(s) have timed out, and starts another one. Once the Stone connects to one of these ShrPcMon processes, it runs correctly, but the extra ShrPcMon processes remain and use resources. (#46928)

Race condition when committing many symbols from multiple gems

When a very large number of new symbols are being created by multiple gem sessions, it was possible for the commands related to SymbolGem communications to intersect in such a way that the Gem is waiting for a response while the SymbolGem is not aware there is work to do. This window is narrow and the problem is rare; but can result in up to a 10-15 second delay in committing. (#46825)

Changing CanonStringBucket's objectSecurityPolicy results in SymbolGem death

AllSymbols and instances of Symbol are protected against changing the objectSecurityPolicy; however the internal buckets that implement the AllSymbols collection were not protected. Changing the security policy for these objects caused the SymbolGem to die with error 2115. (#46410)

Suspended checkpoints resumed after an Epoch GC

When checkpoints have been suspended, if an Epoch GC happened to run during this time window, the suspension was cancelled and checkpoints resumed, without any warning messages in the stone log. If extent copy backups were taking place, which is the usual reason for suspending checkpoints, this could result in the backups being corrupt. (#47133)

Symbol Garbage Collection did not collect Double and Quad ByteSymbols

Symbol garbage collection can be run manually to remove unreferenced Symbols, per the instructions in the System Administration Guide. This code did not identify and mark for removal any symbols that contain characters over 255 (whose class was therefore DoubleByteString or QuadByteString). (#46614)

waitstone may have returned error while stone in startup

When the stone is in startup, it was possible for the waitstone to return an error rather than waiting. (#46518)

Cache warming error when termination due to cache full

When cache warming does not complete because the shared page cache became full, it was logged as an error. This is an expected scenario, so this is now a warning. (#46493)

Keyfile CPU limit not correctly applied for remote gems on large hosts

If the keyfile permits a limited number of CPUs (CPU affinity), but the Stone's machine has fewer CPUs than this limit, remote gems were restricted to this number; remote gems did not use available CPUs on the remote host within the license limit. (#46207)

Keyfile CPU limit does not work correctly with more than 32 CPUs

Keyfile limitations on the number of CPUs did not allow for machines with many CPUs, and would fail if the number of CPUs was 32 or more. The code has been adjusted and can handle up to 1024 CPUs. (#46204)

Upgrade could error attempting to remove documentation category

GemStone upgrade creates temporary categories to install the class comment. It was possible for the removal of this category at the end of the upgrade process to error.

pstack on linux required kernel.yama.ptrace_scope=0

On linux, for pstack to work correctly you may be required to update the kernel parameter kernel.yama.ptrace_scope=0; by default, it may be set to 1. This introduced a security hole.

In v3.4, GemStone's pstack will work with a kernel configuration of kernel.yama.ptrace_scope=1, on distributions using Linux kernel 3.4 or later: Ubuntu 14.04 or later, Redhat 7.x, and SUSE 12. (#46539)

Remote Cache Issues

Remote cache connection behavior

The first gem on a remote node triggers the creation of a remote shared cache, and the creation of a page server on the stone's node. The page server on the stone's node is multithreaded and shared between all gems on that remote node.

If a subsequent gem on that remote node fails to connect to the shared page server on the Stone's node within the timeout of 20 seconds, previously it would create a private pageserver. This could result in excessive page servers on the stone's node. (#46946)

Two new configuration parameters have been added to address this behavior:

STN_GEM_PGSVR_CONNECT_TIMEOUT (page 58) provides control over the timeout specifically for the connection between the remote gem and the page server on the stone's node.

STN_GEM_PRIVATE_PGSVR_ENABLED (page 58), if false, prevents the remote gem from starting a private page server if the connection to the shared page server fails or times out. In this case, the remote gem's login would fail.

Mid-level change used random port to connect to page server on stone's node

The connection between the gem's pgsvr on the mid cache and the gem's pgsvr on the stone cache uses a random port number; it should connect to the well known port number for the pgsvr on the stone cache. (#46382)

System stopZombieSession slow on overloaded remote node

There are code paths in which page manager thread processing can be delayed based on the cache timeout. This can result in operations such as `stopZombieSession:` to take an unreasonably long time for a session on an overloaded remote host. (#46956)

Multithreaded pageserver not shut down after remote cache death

When a remote cache died, the multithreaded page server for that host on the stone's node was not entirely cleaned up. Entries for the page servers continues to use a slot in the shared page cache monitor client table, although they did not have a process table entry. (#47117)

Log file name for remote cache page server used customization

When a remote gem's log file name is defined using NRS directives, and a remote cache was started, the composition of the log file name for the remote cache pageserver incorrectly prepended the remote gem log file name. (#47031)

Risk of SEGV when accessing hidden classes

Sending a message to the results of the private primitive method `Object >> _primitiveAt:` has a risk of SEGV with instances of internal, hidden classes LargeObjectNode or NscNode under some specific circumstances. (#47107)

The new class PrivateObject is now the superclass for internal hidden classes, and sending messages to PrivateObjects other than those implemented in PrivateObject will signal a MessageNotUnderstood rather than SEGV or other undesireable behavior. See "PrivateObject" on page 74.

GsExternalSession>>lastResult may be incorrect with multiple sessions

`GsExternalSession >> lastResult` is used to fetch the results of execution. `lastResult` previously fetched the result from the session that had the most recent previous access, which in an environment with multiple instance of GsExternalSession performing work, could be a different session than the receiver. (#47021)

startnetldi -D did not tolerate GEMSTONE_NRS_ALL with #dir:%D

When the GEMSTONE_NRS_ALL was set to a value that included a #dir:%D, and the NetLDI was started in that environment using the -D argument, it errored and did not start the NetLDI. (#47126)

allSelectors result included duplicates for inherited methods

The results of `allSelectors` could return duplicate symbols, if the superclass implemented the same method. (#46621)

Error on missing UserGlobals dictionary

If UserGlobals dictionary was not present, errors occurred on several methods, including `Behavior>>methodCategories` invoked by GBS browsers. The correction is in the underlying invocation of `GsPackagePolicy currentOrNil`. (#46478)

listReferences failed to find object in large IdentityBags/Sets

If an object is in an IdentityBag or IdentitySet with more than about 1015 or 2030 elements, respectively, a `listReferences:` or `fastListReferences:` operation did not detect the reference. (#46645)

findReferences found references from a large NSC that did not contain the object

If a large UnorderedCollection (NSC) contained an object that referenced a search object, but did not contain the search object directly, the results of a `findReferences:` or `fastFindReferences:` could still have included the NSC, in addition to the correct referencing object. (#47187)

Indexing methods failed to reset ProgressCount

Some indexing processes incremented the ProgressCount statistic but initialized IndexProgressCount. (#45609)

ExecBlock >> selfValue

It was possible for the method `ExecBlock >> selfValue` to return an out of range error, rather than an object or nil. This method is invoked to get process frame contexts by debugger methods, and for GsDevKit continuations contexts. (#46661)

Cannot change objectSecurityPolicy of a DbTransient object

Instances of Classes that are defined as DbTransient can be persisted, but their instance variable data is not written to disk. When setting the objectSecurityPolicy of a committed DbTransient object, the change to the security policy was not visible outside of the session that made the change. (#46655)

transactionConflicts commitResult key was used as details key

The handling of commit transactionConflict keys did not correctly handle synchronized commit failures; the commitResult key was incorrectly used as the conflict details key in recent versions. (#46768)

Now, on synchronized commit failure:

- the commitResult is #failure or #retryFailure, not #synchronizedCommitFailure
- the key to the conflict details is now #'Synchronized-Commit' rather than #synchronizedCommitFailure.

GsSecureSocket could prompt for passphrase

Invoking `GsSecureSocket >> useCertificateFile:withPrivateKeyFile: privateKeyPassphrase:`, with a keyfile that did not require a passphrase and a nil privateKeyPassphrase argument, resulted in a prompt to stdin for the passphrase from within OpenSSL code. (#46913)

Configuration file and parameter Issues

Last line of configuration file without linefeed was ignored

If the last line of a configuration file did not include an end of line indicator (CR or LF), that line was ignored when reading the configuration file. (#46716)

Dynamically set AdminGem config parameters could be lost after reclaim

Before a reclaim operation, the configuration settings are saved, and restored after the reclaim is complete. The ReclaimGem was saving and restoring the complete set of GsUser parameters, not just the ones for Reclaim, which had a risk of overwriting any updates to AdminGem settings. (#46273)

Read-only stone configuration files are not handled correctly

The stone configuration file that is used for extent names must be writable, to allow extents to be added programmatically without creating inconsistency. To avoid risk, the stone should not startup if the configuration file is read-only. This situation is not handled correctly for all cases where configuration files are passed in using the -e and/or -z argument. The problems include unclear error messages and starting up but erroring if an extent is added. (#47054)

String, Character and UTF issues

String hash incorrect in Unicode mode with terminal nulls

If a String ends with characters with codePoint zero, and the repository is in Unicode Comparison Mode, hash was computed incorrectly. (#46932)

Unicode string at:put: handling of invalid index argument

The primitive failure handling code for the `at:` *anIndex* argument was incorrect, resulting in a meaningless error message. (#46537)

Character codePoints could have been truncated by withAll:

When sending `String >> withAll:` with an argument of some particularly structured DoubleByteString argument, codePoints in the result were truncated to less than 256, and the result was an instance of String. (#46879)

Utf8 decodeToString could produce DoubleByteString in String range

If an instance of Utf8 includes encoded Characters with codePoints in the range 128..255, `Utf8 >> decodeToString` produced a DoubleByteString instead of a String. (#46877)

findPatternNoCase:startingAt: error with Unicode string argument

Invoking the method `String>>findPatternNoCase:startingAt:`, with one of the pattern arguments an instance of Unicode16 or other Unicode string class, resulted in an error if the repository was not in Unicode Comparison Mode. (#46975)

Multi-character binary selectors involving a hyphen character required quoting

Symbols containing non-alphanumeric characters (other than underscore) normally require quoting, but this rule does not apply to legal binary selectors (which may contain

only non-alphanumeric characters). Binary selectors with more than one character that included the $- character incorrectly required quoting to evaluate. (#46603)

contentsAndTypesOfDirectory:onClient: incorrect in unicode comparison mode

When the repository is in Unicode Comparison Mode (StringConfiguration is Unicode16), GsFile methods that return file names outside the ASCII range should decode the file names from UFT8 into Unicode strings. The method `GsFile >> contentsAndTypesOfDirectory:onClient:` did not do this correctly when `onClient:` was false. (#46894)

searchlogs script did not respect sessionid, required client in IPv6

The **searchlogs** script returned all entries when the sessionid was used for a filter.

It also did not accept IPv4 addresses for the client filter. (#44458)

GsSocket read: 0 returned nil

The argument for `GsSocket >> read:` is now required to be greater than zero. An argument of zero, which previously returned nil (although no error string was set), will now signal an ArgumentError. (#42322)

GsExternalSession resolveResult:toLevel: broken

This method incorrectly used a 1-based offset for a 0-based C array. (#46919)

Reclaim may be blocked if STN_FREE_SPACE_THRESHOLD lower than #reclaimMinFreeSpaceMb

The system manages reclaim activity vs. free space, by relying on these two settings, to avoid using up all free space when performing reclaim. However, if these two settings are set inappropriately, the system can get stuck where it cannot acquire free space by performing reclaim. In v3.4, the system will generate an error if you attempt to set such a configuration, and in cases where these checks are bypassed, will print a warning in the reclaim gem log.

Hotstandby issues with manually gzipped tranlogs

If a transaction log was manually gzipped before being transmitted (e.g. while the logsender and logreceiver were not connected), on reconnect the transmit would error. Now, manually gzipped tranlogs are read by the logsender. (#46284)

If transaction logs written with record-level compression (using copydbf -c or after being transmitted to the slave by the logreceiver) were manually gzipped, these .gz files were not usable by restore or copydbf. (#46213)

Float passivate-activate resulted in SmallDouble

When a Float in the range of SmallDouble was passivated then reactivated, the result was a SmallDouble rather than the original Float. (#44082)

FileStream errored on read-only files

FileStream could not be used to read files for which the user did not have write permission. (#47155)

FileStream peekTwice failed

The `peekTwice` method failed when sent to a FileStream. (#47156)