GemStone®

GemStone/S Release Notes

Version 6.2

November 2007

GEMSTONE S™

INTELLECTUAL PROPERTY OWNERSHIP

COPYRIGHTS

PATENTS

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

TRADEMARKS

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

Preface

About This Documentation

These release notes describe new features and bugs fixed in the GemStone/S version 6.2 release.

We recommend that everyone using GemStone/S read these release notes before beginning installation or development. These release notes are also available on the GemStone customer website, as described in the next section.

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

Technical Support

GemStone provides several sources for product information and support. The product-specific manuals and online help provide extensive documentation, and should always be your first source of information. GemStone Technical Support engineers will refer you to these documents when applicable.

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

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

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

Help Request allows designated support contacts to submit new requests for technical assistance and to review or update previous requests.

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

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

Downloads and **Patches** provide code fixes and enhancements that have been developed after product release. Most code fixes and enhancements listed on the GemStone Web site are available for direct downloading.

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

TechTips, also in the Learning Center section, provide information and instructions for topics that usually relate to more effective or efficient use of GemStone products. Some Tips may contain code that can be downloaded for use at your site.

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

Technical information on the GemStone Web site is reviewed and updated regularly. We recommend that you check this site on a regular basis to obtain the latest technical information for GemStone products. We also welcome suggestions and ideas for improving and expanding our site to better serve you.

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

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

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

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

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

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

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

Email: support@gemstone.com

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

Requests for technical assistance may also be submitted by email or by telephone. We recommend you use telephone contact only for more serious requests that require immediate evaluation, such as a production system that is non-operational. In these cases, please also submit your request via the web or email, including pertinent details such error messages and relevant log files.

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

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

24x7 Emergency Technical Support

GemStone offers, at an additional charge, 24x7 emergency technical support. This support entitles customers to contact us 24 hours a day, 7 days a week, 365 days a year, if they encounter problems that cause their production application to go down, or that have the potential to bring their production application down. For more details, contact your GemStone account manager.

Training and Consulting

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

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

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

Contents

Chapter 1. GemStone/S 6.2 Release Notes

Chapter 2. Extended Character Set Support

Chapter 3. New TimeZone Subsystem

Chapter 1

GemStone/S 6.2 Release Notes

Overview

GemStone/S 6.2 is a new version of the GemStone Smalltalk object server. This release contains many new features, architectural changes, and fixes for a number of bugs and performance issues.

For details about installing GemStone/S 6.2 or upgrading from a previous release of GemStone/S, see the *GemStone/S Installation Guide*. This release supports Solaris, Linux, and AIX only.

GemStone/S 6.1.6 was a limited release containing a new feature, tools to use tranlog analysis to determine the history of modifications to a data object. This feature is available in GemStone/S 6.2. For more information on this feature, consult the *Release Notes* for v6.1.6 or Appendix H of the *System Administration Guide for GemStone/S* v6.2.

The release contains a bug that affects copydbf of tranlogs across a network (bug 37982). Use ftp to move tranlogs between machines. If this is an issue for your application, please contact GemStone Technical Support.

Changes and New Features

The following changes and new features are available in GemStone/S 6.2.

Updated Documentation

The following manuals have been revised for this release:

- *System Administration Guide for GemStone/S*
- *Gembuilder for C*
- *Topaz Programming Environment*

TimeZone2007 now part of base

GemStone/S now includes a version of TimeZone that supports multiple Daylight Savings Time rules. This is the same as was provided in the TimeZone2007 patch.

The upgrade process is the same with or without the TimeZone2007 patch loaded. If you do not already have the patch loaded, you will need to install a new TimeZone instance after upgrading, and possibly update existing application DateTime instances.

For more details, see Chapter 3, ”New TimeZone Subsystem” on page 35.

New extended character set support

Previous releases of GemStone/S had limited support for characters outside the ASCII and extended ASCII range, and for operations on Strings containing such characters. In particular, sorting and upper/lower case was not handled correctly. Support for case-related operations and collation for all Character sets has been added. For details on this, see Chapter 2, ”Extended Character Set Support” on page 25.

In addition, a number of bugs relating to DoubleByteStrings and DoubleByteSymbols have been fixed. See “DoubleByteString/Symbol related fixes” on page 20.

Architectural changes

The in-band socket is no longer used for any gems. All gems (remote and local) now close their in-band sockets to stone after login completes. Local gems talk to stone using SMC (shared memory communication) via the shared cache. Remote gems use remote SMC; they communicate with their page server on the Stone’s machine via socket, and their page server talks to the stone using SMC.

Gem to gem signalling now works without the gem-stone out of band (OOB) socket. When a gem is sent a signal from another gem, the receiving gem is sent an OOB message telling it to call stone to get the signal and message. Stone will keep a list of up to 50 pending gem to gem signals for any session. After there are more than 50 signals pending, the sending gem will get an error indicating that the receiving session is blocked.

Mid-level Caches

In previous releases, whenever a Gem requested a Page Read, the request was forwarded directly to the pgsvr on the Stone’s machine. With GemStone 6.2, you can set up mid-level caches to reduce the amount of network traffic (Page Read requests) that would otherwise all go to the Stone's machine.

When a Gem needs a page, if it cannot find the page in its local cache, it first looks in the mid-level cache. If the Gem can't find the page in the mid-level cache, it then forwards the request to the Stone.

If a Gem is running on the same machine as a mid-level cache, that Gem will use the mid-level cache as its local cache.

GemStone provides several methods in class System that let you connect to, and obtain information about, the mid-level caches on your system.

Connection Protocol

Class methods in the Shared Cache Management category allow you to connect to a midlevel cache.

`midLevelCacheConnect:` *hostName*
Attempts to connect to a mid-level cache on the specified host, if the cache already exists. The session's Gem process must be on a machine different from the machine running the Stone process.

`midLevelCacheConnect:` *hostName* `cacheSizeKB:` *aSize* `maxSessions:` *nSess*
If a mid-level cache does not already exist on the specified host, and *aSize* > 0, attempts to start the cache and connect to it. If a cache is already running on the host, this method attempts to connect to the cache and ignores the other arguments.

The size of the mid-level cache is controlled by the method argument *aSize*, rather than configuration parameters (as with other shared caches).

Reporting Protocol

Class methods in the Shared Cache Management category return lists of the shared caches on your system.

`remoteCachesReport`
Returns a String that lists all shared caches that the Stone process is managing, not including the cache on the Stone machine.

`midLevelCachesReport`
Similar to `remoteCachesReport`, but only includes the mid-level caches.

New PageManager Process

The Page Manager Gem process has been introduced into GemStone/S. This process is used in GemStone/S 64 Bit and is valuable for offloading the Stone in configurations including remote caches.

When pages need to be disposed, it is necessary to communicate with all remote caches in order to ensure that the pages are not in use anywhere, before the pages are actually made available for reuse. The Page Manager Gem addresses this problem by taking the list of potential pages to be disposed, and checking if any of them are in use in the local cache. It then communicates with each of the remote shared cache page servers (if any), which provide the same information for the remote caches. The set of pages that are not in use and can be safely disposed is returned to the Stone, which performs the actual page disposal. The Page Manager Gem remembers any pages that are in use, and will retry the removal at a later time.

The Page Manager Gem is also responsible for starting up remote page caches, and handles remote page cache shutdown or unavailability.

The Page Manager Gem is always running. It is started by the Stone during Stone startup, and shut down by the Stone during Stone shutdown. You cannot shut down the Page Manager Gem while the Stone is running; killing the process also brings down the Stone. It does not run during crash recovery or restore from tranlog situations, but is automatically started when recovery or restore is complete. The Page Manager Gem does not reference a commit record, and has no transactional view of the database. It therefore does not affect the commit record backlog.

Maximum number of remote caches increased to 1024

Previously, the limit on the number of remote caches was about 500. This has been increased, allowing for more remote caches and aio page servers.

Compile and link changes

This release uses updated compilers. A C++ compiler is now required. See the *GemStone/S Installation Guide* for required compiler versions.

For updated compile and link commands, see the updated *GemBuilder for C* manual.

Session control changes

The process of terminating sessions, both the way the system handles programmatic termination of sessions and automatic termination of unresponsive sessions, has changed in this release.

In addition to the process changes, some special cases have been fixed to make this more robust, new methods have been added, and several bugs have been fixed.

Terminating gems

The way session termination is handled has been redesigned in this release. This in particular affects terminating sessions using System class >> stopSession and when STN_GEM_TIMEOUT is exceeded. This is the new behavior:

1. An OOB signal is sent to the gem, requesting it to stop.
2. The gem is given 60 seconds to stop on its own.
3. If it does not stop within this timeout, the gem's cache slot (and the slot of it's page server, for remote sessions) is poisoned (any access will cause the gem to be killed).
4. The stone then starts the process of logging out the gem.

   For local sessions, the gem's logout can only be completed immediately if the stone receives a disconnect on the gem's OOB socket, or if the gem's process no longer exists. For remote sessions, the gem's page server process must have terminated, and either the disconnect received, or the remote cache for the gem shut down.
5. If the stone isn't able to complete the logout, the gem's commit record is dereferenced.
6. After 60 seconds, if the stone still cannot complete the logout, the gem will be killed via kill -TERM. For remote session, the session’s pgsvr is sent a kill -TERM. Terminating process this way will not result in stuck spin locks.

   Gems are never sent a kill -9, to avoid the risk of stuck spin locks.

New method to terminate session immediately

The method System class >> terminateSession: *sessionId* has been added. This performs the same process as System class >> stopSession, with the difference that in there is no timeout as in step #2; the session is immediately logged out. While termination is immediate, there may be some delay due to the following steps that clean up session state.

New method to prevent OS process kill

The method System class >> disableTerminationSignals has been added. This can be sent on a per-session basis to avoid gem processes receiving the kill -TERM done in step #6. This does not prevent pgsvr processes for remote gems being killed.

Use of this option has the risk of the system accumulating zombie processes. This does not cause a functional problem but may use up resources. The primary intention of this option is when linked and RPC sessions are running in the same client process; in this case, terminating a linked session will also terminate the RPC sessions.

Lost OT handling

After the ABORT_ERR_LOST_OT_ROOT signal is sent to sessions that are referencing the oldest commit record, the sessions are handled as follows:

If STN_GEM_LOSTOT_TIMEOUT = -1, then the following occurs immediately:

- Retracts the session's commit record
- Poisons its cache slot (the session will get a fatal error if it references the shared cache), and the cache slot of the pgsvr for remote sessions.

When the parameter setting is -1, the session is not terminated via kill -TERM.

If STN_GEM_LOSTOT_TIMEOUT >= 0, the stone waits for STN_GEM_LOSTOT_TIMEOUT seconds for the session to respond to the signal (by aborting, begin transaction, or logging out). If the session does not respond, it continues with the lost OT processing below. If STN_GEM_LOSTOT_TIMEOUT = 0 (a timeout of 0), then the following lostOT processing occurs immediately:

- Retracts the session's commit record
- Poisons its cache slot (the session will get a fatal error if it references the shared cache).
- Forcibly log off the session, if possible (see “Terminating gems” on page 4).
- If the gem does not stop within STN_GEM_TIMEOUT, the session is sent kill -TERM. Sessions are never sent a kill -9, to avoid the risk of fatal stuck spin locks.

Sessions that are performing reclaimAll or that are logged in as Nameless are not forcibly logged off or sent kill -TERM; the session’s commit record is retracted and the cache slot is poisoned.

Sessions can now log themselves out

The method `System Class >> logout` has been added, to allow sessions to log themselves out even if they do not have session control privilege.

Cache Statistics Changes

GCSI

GemStone/S has added the ability to write C programs to read cache statistics directly from the shared page cache, avoiding the need to log in a gem session. For details, see the updated GemBuilder for C manual.

Statmonitor and VSD changes

The way cache statistics are recorded has changed with this release. You can not use older versions of VSD to read files produced by statmonitor in version 6.2. You must use the version of VSD (version 3) provided with 6.2. VSD version 3 will read statmonitor files produced by earlier versions of GemStone/S.

In addition, the VSD template files have been corrected and updated.

DateTimes now support millisecond resolution

While DateTime has included internal millisecond values, the official support was to the seconds level. This has been modified so DateTime operations are fully supported to millisecond resolution.

The following methods have been added:

```
DateTime >> addMilliseconds:
DateTime >> asMillisecondsGmt
DateTime >> asStringGmtMs
DateTime >> asStringMs
```

Non-Smalltalk time output format change

The date portion of the time output format has changed. This includes the results of the topaz time directive, and the log file header output. For example, in 6.2 the output is:

```
11/05/07 12:05:09.826 PST
```

Previously, this would have been:

```
Mon 05 Nov 2007 12:05:09.826 PST
```

Restore from Backup protocol now logs off

Changes in object management in version 6.2 introduce risk during restore operations. As a result, following execution of restore from backup and restore from log methods, the session will be logged out, and must log in again to continue. The following methods, as well as any other methods that call these methods, are affected:

```
commitRestore
restoreFromArchiveLogs
restoreFromBackup:
restoreFromBackups:
restoreFromCurrentLogs
restoreFromLog:
```

Backups now permitted when specialized GcGems running

Making backups while any combination of GcGems is running is now supported. In GemStone/S 6.1.5, making a GemStone backup of the repository was disallowed while any of the specialized GcGems were running - the Reclaim GcGem, Epoch GcGem, Parallel Dead Reclaim or Parallel Shadows Reclaim GcGems. In release prior to 6.1.5, making backups under these circumstances risked the backup hanging (bug #29567).

GcGem statsEnabled parameters removed

The GcGem configuration parameters #reclaimStatsEnabled and #epochGcStatsEnabled, and the associated variables #reclaimStats and #epochGcStats are no longer used or supported. These variables were awkward to use and caused a number of problems. Collecting of these statistics had been disabled previously while running PSR GcGems and during reclaimAll (Bugs 30429 and 29980, fixed in v6.1.5).

New cachewarming utility

A new utility script has been added to start shared page cache warming gems.

Each cache warming gem reads a portion of the object table into the shared page cache. Cache warming gems automatically terminate when the shared cache becomes full or their portion of the object table has been completely loaded, whichever happens first. Cache warmers run outside of transaction and respond to sigAborts. They are safe for use in production.

```
startcachewarmer [-n<numGems>] [-p<password>] [-s<stone>]
    [-u<userID>] [-w<delayTime>][-W]
    -h  Display this message and exit
    -n<numGems>The number of gems to start (default: 1)
    -p<password>GemStone password for logging in gems (default: 'swordfish')
    -s<stone> Specify the name of running stone (default: 'gs64stone')
    -u<userID> GemStone user for logging in gems (default: 'DataCurator')
    -w<delayTime>Wait the specified number of seconds between spawning gems (default: 1)
    -W  Wait for cache warming gems to exit before exiting this script (default: spawn
        gems in the background and exit immediately)
```

RcQueue improvements

RcQueueEntry

A new class has been added, RcQueueEntry. This is a subclass of RcQueueElement and is used interchangeably. After upgrading, any objects added to an RcQueue will be instances of RcQueueEntry, rather than RcQueueElement. There is no need to perform any modifications to your existing RcQueues.

The new RcQueueEntry class keeps two timestamps: seconds (since 2005) and microseconds. This allows much greater precision in sequencing the objects in an RcQueue. In addition, it avoids any possible problems when the SmallInteger range is exceeded in 2012; see bug 22507, 'RcQueue related bugs'.

Note that RcQueues are only reduced-conflict when the configuration parameter CONCURRENCY_MODE is set to #NO_RW_CHECKS. A setting of #FULL_CHECKS will result in read / write and write / read conflicts.

Performance

For performance, conflicts on the RcQueue itself are no longer handled by reduced-conflict (RC) retry logic. This means attempts to grow the RcQueue while in use are likely to fail. Therefore, an RcQueue is no longer lazy initialized - session components are added at RcQueue creation time when the new: method is used. It is advisable, if you know the maximum number of sessions that will be using the RcQueue, to specify the size on creation.

An improvement was made in the algorithm for detecting the first element in an RcQueue; see “Inefficient search code in RcQueue” on page 19.

Also, several methods have been reimplemented in C for performance.

Added methods

`RcQueue >> removeIntoArray:` anArray

Removes the leading element from the receiver and stores the element in anArray at index 1. A SmallInteger representing the session ID of the session that added the element to the queue is also stored in anArray at index 2. If the receiver is empty, returns nil.

`RcQueue >> removeIntoArray`

Removes the first element from the receiver and stores the element in a new Array at index 1. A SmallInteger representing the session ID of the session that added the element to the queue is also stored in the new Array at index 2. If the receiver is empty, returns nil.

`RcQueue >> removeObject:` anObject `addedBySessionId:` sessionId

Removes the given object from the receiver assuming it is the oldest object added by the given session Id. Returns the object or nil if the object was not found or the receiver is empty.

RcQueue no longer usable with continueTransaction

These optimizations circumvent the process of safely handling continueTransaction. To avoid risk, updating an RcQueue after executing a continueTransaction is no longer allowed.

After executing continueTransaction, if you attempt to update an RcQueue, the new error RT_ERR_RC_UPDATE_DISALLOWED is raised, and you will not be able to commit the transaction; you must abort.

The following method has been added:

`System class >> inContinueTransaction`

Ability to turn off handling of terminating OS signals

A new environment variable has been added, GS_DISABLE_SIGNAL_HANDLERS. When this environment variable is enabled (by setting it to any value; the actual setting does not matter) in the environment for a gem process, this gem sessions will not attempt to handle certain types of signals, but will shut down immediately. This avoids side effects with user action or client Smalltalk code.

When the environment variable GS_DISABLE_SIGNAL_HANDLERS is set, and the session receives a SIGSEGV, SIGBUS or SIGILL signal:

- Process will immediately exit
- Core files will not be generated
- C call stacks will not be printed

The applies only to gem sessions; other GemStone processes behave as before.

If this environment variable is enabled, a notice is printed in the header of the gem log.

Locale Added

A class Locale was added in GemStone/S 6.1.5, providing operating system locale information in GemStone. This class was inadvertently not documented in the *Release Notes* for 6.1.5.

The Locale class provides all the information that is provided by the OS locale; see the image and OS documentation (man localeconv) for more information. GemStone currently only uses the decimalPoint setting, to provide localized reading and writing of numbers involving decimal points. Note that Smalltalk syntax requires the use of '.' as the decimal point separator, so expressions involving literal floating point numbers will still require use of the '.', regardless of Locale.

The Locale class has a new feature in version 6.2; you may now override the OS locale information, using:

`Locale Class >> setCategory:` *categorySymbol* `locale:` *LocaleString*

Valid categories are:

#LC_CTYPE	locale's character handling
#LC_NUMERIC	locale's decimal point handling
#LC_TIME	locale's date and time handling
#LC_COLLATE	locale's collation setting
#LC_MONETARY	locale's monetary handling
#LC_MESSAGES	locale's messages handling
#LC_ALL	all locale categories

See the OS man page for setlocale for more information.

For example, to set GemStone to use decimal localization appropriate for Germany, use:

```
Locale setCategory: #LC_NUMERIC locale: ‘de_DE’.
```

To set up to use UNIX default values:

```
Locale setCategory: #LC_ALL locale: ‘C’.
```

New methods have been added to provide a way to convert a String to a Float without consideration of the current Locale setting:

```
Float >> fromString:useLocale:
DecimalFloat >> fromString:useLocale:
```

New methods to return platform Byte Order

To determine the native byte order of the platform that the gem or stone are running on, new methods have been added:

```
System class >> gemIsBigEndian
System class >> stoneIsBigEndian
```

Added ability to manually disable a user's account

A new method was added, `UserProfile>>disable`. This allows a userProfile to be disabled programmatically; the login failure reason will show 'DisabledByAdmin'. To re-enable the UserProfile, set the password using the password: method.

GemStone stacks now available on AIX

GemStone/S on AIX now has the ability to capture C and Smalltalk level stack traces, as pstack does on Solaris and Linux. This allows kill -USR1 to print the C and Smalltalk level stack traces to the process log file.

Improvements in logging

More details on lostOTRoot handling in stone log

The lostOT behavior has been changed, so it is always reported in the stone log when a session's view is pulled. Previously this was only done if a session was killed, but not if the resources were pulled but the session was not killed.

More precise log timestamping

Timestamps in the stone log are now printed to the nearest millisecond, if no message was printed within the last 10 milliseconds.

Additional information on topaz login/logout

Topaz prints additional information about the session on login and on logout.

Backup start time now printed in log

Previously, only backup completion time was noted in the stone log. Now both backup start and completion are logged.

More details on shutdown

The information printed on stone shutdown now includes details on the session that performed the shutdown.

Improved printing of very long Strings

Printing of very long strings is truncated to avoid overflow problems. This truncation has been improved to avoid the results when there are embedded String demarcation characters.

The following methods have been added:

```
PrintStream >> maxSize
PrintStream >> isFull
PrintStream >> overflow
PrintStream class >> printingOn:maxSize:
```

String >> printString now call PrintStream class >> printingOn:maxSize:, and will cleanly print 1000000 (1M) bytes of data. To override this, use the method printStringWithMaxSize:.

Installation directory permissions

Formerly, the installgs script modified the operating system directory permissions for $GEMSTONE/data and $GEMSTONE/ualib, to include world write. This has been restricted, to group write. If you use installgs, verify the resulting directory permissions are as required for your application usage and security requirements.

Updated zlib libraries

The zlib libraries used to do the compression / decompression of .gz files have been updated to the latest version, which is 1.2.3.

Optimization of or: and and:

Previous versions optimized _and: and _or: to simplify the block, while and: and or: were unoptimized message sends. Now, and: and or: are optimized the same as _and: and _or:. It is no longer useful to use the underscore versions of these methods.

ANSI exception handling

GemStone Smalltalk now implements the ANSI Exception handling protocols. Provision is made for signaling that an exception has occurred and for defining handlers for signaled exceptions.

The legacy exception handing mechanisms are still available; upgraded applications do not need to be modified. The ANSI framework is built completely out of the legacy framework, and is intended to be backwards compatible with it. In order to accommodate the legacy framework, the top-level exception in the ANSI framework is named `ExceptionA` rather than `Exception`.

ANSI Exceptions are class-based, meaning that you use a class in the ExceptionA hierarchy to describe errors and other exceptions in your GemStone Smalltalk programs. ANSI errors, for example, include the new GemStone Smalltalk classes `MessageNotUnderstood` and `ZeroDivide`.

SUnit, an open source testing framework, has also been added.

The following classes have been added in support of ANSI exception handling and the SUnit testing framework:

```
Error
ExampleSetTest
ExceptionA
ExceptionHandler
ExceptionSet
FailedMessage
MessageNotUnderstood
Notification
ResumableTestFailure
ResumableTestFailureTestCase
SimpleTestResource
SimpleTestResourceTestCase
SUnitDelay
SUnitNameResolver
SUnitTest
TestCase
TestFailure
TestResource
TestResult
TestSuite
Warning
ZeroDivide
```

changeClassTo: restrictions removed

The method changeClassTo: previously only permitted changing the class of an instance to a subclass of its class. These restrictions have been relaxed. The new rules are:

- The receiver's class must have the same implementation as the new class (bytes, pointers, or non-sequenceable collection).
- If the new class is a kind of IdentitySet, then the current class must also be a kind of IdentitySet.
- Sets and Bags may not be changed from one class to the other, and IdentitySets and IdentityBags similarly may not be changed from one class to the other.
- The new class must not be a kernel class for which instance creation is disallowed.

Please use caution in changing the class of an object; injudicious use of this method can cause problems in your application.

Methods that now perform aborts

The following methods now automatically perform an abort. If there are modifications to persistent objects that would be lost due to the abort, the method fails to abort and returns the new error 2412, `#rtErrAbortWouldLoseData`.

The following methods are affected by this change. Methods that call any of these methods are also affected. This is not an exhaustive list.

```
Repository >> fullBackupTo:MBytes:compressed:
Repository >> continueFullBackupTo:MBytes:
Repository >> continueFullBackupCompressedTo:MBytes:
Repository >> findObjsConnectedTo:
Repository >> listReferences:
Repository >> listReferences:withLimit:
Repository >> markForCollection
Repository >> markForCollectionWait:
Repository >> markGcCandidates
Repository >> reclaimAll
Repository >> repairWithLimit:
System class >> findObjectsLargerThan:limit:
UnorderedCollection >> createIdentityIndexOn:commitInterval:
UnorderedCollection >> createEqualityIndexOn:commitInterval:
```

Other added methods

The following methods have been added to the image:

```
Character >> codePoint
Character class >> codePoint:
Class >> allSubclasses
Class >> subclasses
Collection >> allSatisfy:
Collection >> anySatisfy:
PositionableStream >> nextLine
PositionableStream >> peekFor:
System Class >> voteState
```

Topaz changes

Updated documentation

The *Topaz Programming Environment* manual has been updated for this release.

EXPECTVALUE additional argument specification

EXPECTVALUE has additional argument specification allowing OOPs as well as objects to be used:

%@*OOPOfClass* specifies an instance of the class that has the OOP *OOPOfClass*.

/@*OOPOfClass* specifies an instance of the class that has the OOP *OOPOfClass*, or an instance of any of its subclasses.

Added topaz commands

The following commands have been added in this release. For more details on these commands, see the updated *Topaz Programming Environment*.

NBRUN

Similar to printit, but execution is nonblocking, so the application can proceed with non-GemStone tasks while the expression is executed. To get the results of the execution, see nbresult.

NBRESULT

Wait for and display the result of a previous nbrun call. This call must be preceded by a set session to switch to the session of an outstanding nbrun.

Previously available commands

The following commands were available in previous releases but not documented in the *Topaz Programming Environment* manual. For more details on these commands, see the updated *Topaz Programming Environment*.

DISASSEM

Disassemble the specified GsMethod, displaying the assembly code instructions.

GCITRACE

Turns GCI tracing on

PROTECTMETHODS

After this command, all subsequent method compilations during the current session must contain either a <protected> or <unprotected> directive.

UNPROTECTMETHODS

Cancels the effect of protectmethods,

Changes in Errors

The following errors have been added in the 6.2 release.

Error	Error Name	Description
2400	RT_ERR_CANNOT_SUSPEND_LOGINS	Cannot suspend logins. Args:(1) reason
2407	RT_ERR_NOREPLICATES_WITH_MID_CACHE	Replicates not supported when using a mid-level cache
2412	RT_ERR_ABORT_WOULD_LOSE_DATA	A method is being run that requires an abort, however, an abort would result in lost data as there are modified objects.
2500	RT_ERR_RC_UPDATE_DISALLOWED	An attempt was made to update an RC object in a continued transaction. The transaction must abort.
4052	GS_ERR_SIGTERM	gem or topaz -l process exiting because SIGTERM was received
4148	GS_ERR_NO_FREE_FRAMES	

The following errors have new numbers in version 6.2. The three BKUP_ERR_RESTORE_* errors are now fatal errors; restore now terminates the session.

Error Name	GemStone/S 6.1.x	GemStone/S 6.2
BKUP_ERR_RESTORE_LOG_FAIL	3012	4049
BKUP_ERR_RESTORE_LOG_SUCCESS	3011	4048
BKUP_ERR_RESTORE_SUCCESSFUL	3008	4046

Error 2330, #RT_ERR_INVALID_RESTORE_TIME, has a modified error message which includes both the current restore status time and the argument time, including time zones. For example:

```
Repository already restored to 09/20/07 15:21:58 PDT , which is
after specified time 09/20/07 15:00:00 PDT
```

Changes in Cache Statistics

The following statistics have been added in the 6.2 release:

NumInPgsvrWaitQueue
The number of remote sessions logged out and waiting for their pgsvr process to die.

NumInRemoteKillQueue
The number of sessions that the page manager is in the process of killing.

NumInRemotePidQueryQueue
The number of sessions for which the page manager is trying to determine existence.

NumInSetLostOtQueue
The number of sessions waiting for set lostOt in shared caches.

OldCommitRecordPagesWrittenByGem
The number of old commit record pages in the cache because of a write done by a gem. An old CommitRecord page is an commit record from a previous release of GemStone.

OldCommitRecordPagesWrittenByStone
The number of old commit record pages in the cache because of a write done by the stone. An old CommitRecord page is an commit record from a previous release of GemStone.

OldestCrSessNotInTrans
The session ID of the oldest session that is not in a transaction that is currently referencing the oldest commit record. This session may be preventing the commit record from being disposed.

PageMgrPagesNotRemovedFromCachesCount
The total number of pages the Page Manager was unable to remove from one or more shared page caches.

PageMgrPagesPendingRemovalRetryCount
The current number of pages that could not be removed from shared page caches by the page manager on the first attempt and are waiting to be retried.

PageMgrPagesReceivedFromStoneCount
The total number of pages the Page Manager session received from the stone to remove from shared page caches.

PageMgrPagesRemovedFromCachesCount
The total number of pages the Page Manager has successfully removed from all shared page caches.

PageMgrRemoveFromCachesCount
The total number of times the Page Manager has attempted to remove page s from shared page caches.

PageMgrRemoveFromCachesPageCount
The total number of pages the Page Manager has attempted to remove from shared page caches. This statistic includes pages processed by page removal retry operations, which occur whenever a page cannot be removed from a shared page cache on the first attempt.

PageMgrRemovePagesFromCachesPollCount
The number of times the page manager called poll() or select() to determine which

cache page servers have completed removing pages from their shared caches. This statistic represents the value during the most recent page disposal operation and is not cumulative. It will always vary between zero (when there are no remote shared caches on the system) and the number of remote shared page caches.

PageMgrTimeWaitingForCachePgsvrs
The total amount of real time in milliseconds the page manager has spent waiting to receive data from remote cache page servers.

PagesNeedRemovingThreshold
Threshold for page manager to process the backlog described by PagesWaitingForRemovalInStoneCount.

PagesNotFoundInCacheCount
The total number of pages not found in the shared cache when the Page Manager gem or cache page server attempted to remove them.

PagesRemovedDirtyFromCacheCount
Number of dirty pages successfully removed from the cache by the cache page server or the Page Manager at the stone's request.

PagesWaitingForRemovalInStoneCount
The number of pages in the Stone that are waiting to be removed from the Shared Page Cache by the Page Manager.

PgsvrPid
Process ID of the session's pageserver (remote sessions only).

PinnedPagesCount
The number of pages the process has pinned (locked) in the shared cache. Pages may be pinned by more than one process at the same time.

PinnedPrivatePagesCount
The number of pages the process has pinned (locked) in its private page cache.

RemoteCachesNeedServiceCount
The number of outstanding requests to start or stop a remote shared page cache. Requests are initiated by the stone and executed by the Page Manager session.

RemoteSharedPageCacheCount
The total number of remote shared page caches attached to the system.

SmcQueueSize
The number of sessions in the SMC (shared memory communication) queue waiting to be added to the run queue by stone.

TimeInStonePageDisposal
The total amount of real time in milliseconds the stone has spent performing page disposal tasks.

WaitsForOtherReader
PageRead operations avoided by waiting for read already in progress by another process.

Changes in Configuration Parameters

Default spin lock count increased

The default spin lock count was increased to 4000 for multi CPU systems.

Pages now detached on commit and abort

The configuration options GEM_DETACH_PAGES_ON_COMMIT and GEM_DETACH_PAGES_ON_ABORT have been removed. Pages are now always detached on commit or abort.

STN_GEM_TIMEOUT

This setting is used as a gem login timeout. Now, if it is set to 0 (the default, meaning wait forever), a gem login timeout of one minute is used. Previously it was five minutes.

GEM_IO_LIMIT updated

The parameter GEM_IO_LIMIT was not functional in a number of past releases (see "GEM_IO_LIMIT configuration parameter ignored" on page 18). This has been fixed, and the default has been changed to 10000.

Added configuration parameters

The following configuration parameters have been added:

GEM_DBF_FILE_LOCK

When true, the Gem locks the extents when opening the extents during login.

Default: FALSE

GEM_PGSVR_UPDATE_CACHE_ON_READ

Determines the read behavior of the gem's remote page server when pages are read from disk. If this option is set to TRUE, pages read from disk are also added to the shared page cache on the page server's host. If this option is FALSE, pages read are not added to the page server's shared cache. Has no effect for gems local to the repository extents (local gems do not have a page server).

Runtime equivalent: #GemPgsvrUpdateCacheOnRead
Default: FALSE

STN_PAGE_REMOVAL_THRESHOLD

Minimum batch size for the page manager system gem. When the number of pages waiting to be processed by page manager is greater than this value, then the page manager will request the pages from the stone and process them. Otherwise the page manager will wait until this threshold is exceeded before requesting pages from the stone. The stone cache statistic PagesNeedRemovingThreshold reflects the current value of this parameter.

Runtime equivalent: #StnPageRemovalThreshold
Default: 40
Min: 0
Max: 1792

Bugs Fixed

The following bugs in GemStone/S 6.1.5/6.1.6 have been fixed in GemStone/S 6.2:

Remote sessions may have received multiple LostOT signals on login

If a remote gem becomes unresponsive, receives a lostOT, but does not shut down, its page server may remain, waiting on I/O from to unresponsive gem. If the session id was reused, the system may have incorrectly matched this stale page server rather than the page server for the new session, and treated the new session as having the lostOT. A secondary problem caused a loop, which resulted in the multiple lostOT signals. (#36725)

Risk of stack corruption on asynchronous error handling in user action code

If an asynchronous signal (such as a sigAbort) is handled while returning from user action code, results may be incorrectly added to the call stack, resulting in a corrupted stack.(#35922)

GEM_IO_LIMIT configuration parameter ignored

Changes to the value of GEM_IO_LIMIT in the configuration file, or using runtime configuration parameter #GemIOLimit, had no effect. (#19327)

Number related bugs

Integer raisedTo: returned Float

This method always returned a Float. The ANSI specification states that for an Integer argument, Integer >> raisedTo: should return the same type as the receiver, an Integer. (#37320)

Locale not used in ScaledDecimal read/writes

ScaledDecimal read and write operations did not use the Locale decimalPoint setting. (#36667)

Poor performance in Fraction comparison

Fraction comparisons using < now use a faster algorithm, providing much better performance. (#36653)

asFloat incorrect results for some large Integers

For some large Integers that are outside the range that can be exactly represented as Float values, on conversion the rounding was incorrectly rounded down rather than up, as required per IEEE 754. (#36251)

Very small Floats converted to Fractions produced incorrect results

Very small Floats (Floats that are very close to zero), when converted to Fractions, either produced zero or a Fraction that was not exactly equivalent to the original Float. (#36173)

ScaledDecimal product scale depended on ordering of multiplicands

When two ScaledDecimals are multiplied, the resulting ScaledDecimal scale was the scale of the first multiplicand. Now, the product's scale has the larger of the two scales. (#33818)

bitShift: incorrect results for negative shifts >= 32

SmallInteger>>bitShift: returns the receiver, rather than the correct answer of 0, for negative shifts larger than 31. (#33891)

Activate/Passivate of structures with ScaledDecimals or Fractions with Large components

When structures containing ScaledDecimals or Fractions with components that were LargePositiveIntegers or LargeNegativeIntegers were passivated and activated, it resulted in errors or incorrect results. (#35805)

Passivate/activate sensitive to Locale changes

The passivation mechanism used the current decimal separator, resulting in errors if Floating point values were activated in a different Locale than the one in which they were passivated. (#36540)

RcQueue related bugs

RcQueues would have failed after January 5, 2012

Due to a SmallInteger constraint on RcQueue internal component RcQueueElement, adding items to an RcQueue would have failed after January 5, 2012, when the number of seconds since January 1, 1970 exceeded the SmallInteger range. (#22507)

This has been avoided by the introduction of a replacement internal structure, RcQueueEntry. See “RcQueueEntry” on page 7.

Inefficient search code in RcQueue

The code in RcQueue that implemented in binary search to find the next element included inefficient code. (#36055)

Commits on RcIdentityBag could lose data

Under specific circumstances when resolving transaction conflicts, depending on the position of the conflicting changes within the internal collection’s internal structures, it was possible for modifications to these objects to be lost. (#36040)

Processing of “Zombie” objects could leave stuck invisible exclusive locks

Under certain rare conditions, the system can generate "zombie" objects, which are flagged as dead, but which still have entries in the object table. In GemStone/S 6.0, code was added to detect and clean up these zombies, but failed to clear the exclusive lock on the object, leaving it stuck "on". If the zombie object’s OOP is reused for a newly created object, and the session commits this object, the commit would fail with #Read-ExclusiveLock transactionConflict. (#34049)

Stone could run hot following problems with remote gem

If a remote gem terminated abnormally, there was a risk that the stone would run hot while the page server was performing cleanup. (#36352)

DoubleByteString/Symbol related fixes

String >> asDoubleByteString fails on strings with leading nulls

Strings with leading nulls, when converted to DoubleByteStrings, resulted in DoubleByteStrings with the nulls incorrectly at the end of the string (#37015)

Converting large String/Symbol to DoubleByte caused corrupt result

DoubleByteString >> withAll: or addAll: with a String or Symbol argument that is larger than 4048 produced corrupt results. (#33569)

DoubleByteString copyFrom:to:into:startingAt: broken for String destinations

DoubleByteString>>copyFrom:to:into:startingAt: with a destination collection of type String produced corrupt results. (#33539)

<=, >=, and sort incorrectly case-sensitive for DoubleByteStrings

String sorting by default is case-insensitive. However, DoubleByteString inherited implementations of <= and >=, which were case sensitive. As a result, comparisons of DoubleByteString, and sorting of collections that included DoubleByteStrings, may have been incorrect. This included the default sort of SortedCollection. (#33404, #33405)

Issues in compiling methods with DoubleByteString source

The fix for 33365 (Invalid and noncanonical DoubleByteSymbols) in 6.1.5 did not entirely address the problems in handling DoubleByteStrings and Symbols within compiled methods.

bug 33954 (Compiling a method that is a DoubleByteString turns all literal strings into DoubleByteStrings) still occurred under some conditions. In addition, where the Characters over 255 occurred in the latter part of a long method text, they may not have been correctly converted, resulting in a SIGSEGV (bug 24456). These bugs were fixed in version 6.1.6.

There are further related fixes in this release, for the risk of object does not exist errors during method complication. (#37175).

Possible invalid or noncanonical DoubleByteSymbols

Bug 33365 (Invalid and noncanonical DoubleByteSymbols) was fixed in version 6.1.5. However, there are certain circumstances and platforms under which it was still possible to create DoubleByteSymbols containing entirely characters with values under 128, and noncanonical DoubleByteSymbols. This bug was fixed in version 6.1.6. (#34028)

Character >> isAlphaNumeric incorrect for characters 192-255

Characters in the range 192 to 255 contain roman letters with accents, umlauts, circumflex, etc. These Characters were incorrectly returning false when sent the message isAlphaNumeric. (#34720)

Incorrect comments for Character comparison methods

The comments for the comparison methods in Character (>, >=, <, <=) indicated that they compare by Unicode value. Actually, they use the case-insensitive collation order. The comments now reflect the behavior. (#36224)

DoubleByteString >> wordAt: broken

The method DoubleByteString >> wordAt: was missing a parenthesis and caused a walkback. (#36132)

DoubleByteString >> asNumber truncated decimal values

Converting a DoubleByteString containing a floating point value to a number created an Integer with the decimal part truncated. (#36732)

PositionableStream >> upToEnd reset position

The method PositionableStream >> upToEnd incorrectly reset the position to the start of the stream. (#34584)

Migrating instances of byte classes (other than CharacterCollections) failed

Object migration contains code to handle migration of instances of Byte classes. This code failed for instances of byte classes other than CharacterCollections and its subclasses (String, etc.) (#29778)

Class >> classHistory: now restricted

It was possible to set the classHistory of an invariant instance of Class. It should not be possible to modify invarient objects. (#37438)

Error messages reported large oops as negatives

In error messages that include the OOP of an object, such as 2115 #authErrSegRead and 2101 #objErrorDoesNotExist, if the OOP was a LargeInteger, the error message incorrectly reported a negative value. (#35640)

Tranlog analysis scripts limited number of tranlogs per operation

The number of tranlogs that the tranlog analysis scripts could process at once was limited by command line size limitation and file descriptor limits. These problems have been addressed; the limitation remains that no more than 256 tranlog files can be processed in one operation. (#35947)

Nonblocking logins had NetPoll() problem

A call to NetPoll() was being made to detect errors during nonblocking login. This could consume a socket read, resulting in socket errors.

NonBlocking logins are not used by GemStone or GBS; they are used by GBJ, or may be called specifically using the GemStone method GciInterface>>nbLogin: or the GemBuilder for C call GciNBLoginEx(). (#37931)

findDisconnectedObjects not protected against other GC operations

FDC does not acquire the Gc lock. If another GC operation executed while a Repository >> findDisconnectedObjects was running, the results returned from the findDisconnectedObjects may have been inconsistent. (#30915)

PageType documentation confusing

In the System Administration Guide and in method System >> cacheStatisticsDescription, the PageTypes are provided as indexes into an Array returned by the statistics. This was not the same as the actual Integer PageType, which is 0-based. For example, A data page is PageType 4, but in slot 5 of the Array returned. This has been clarified in the comments (#37178)

Object audit re-scan reported twice as many errors

After the initial scan that objectAudit performs, if errors are found a detailed re-scan is performed. This did not reset the error count to 0, so the final reported error count was twice as large. (#36771)

Also, the object audit report has been enhanced to print if it is run in single-user mode and all checks would be performed, or in multi-user mode when only partial checking is done.

Spin lock issues

Kill -9 could cause page cache faults

If a gem holding a spin lock was killed using kill -9, and the spin lock was recovered (avoiding cache shutdown), the cleanup may not have been complete, which could result in shared page cache corruption. (#32770)

Stuck spin lock could have caused stone to hang

If a session is terminated while holding a spin lock, in some cases the shared page cache monitor can recover the spin lock, and the system can continue running. However, in cases where the spin lock is not recoverable, if the stone attempted cleanup before the shared page cache monitor, there was a risk that the stone would hang. (#34772)

Collection deepCopy performed do:

The method Collection >> _deepCopyWith:, called by deepCopy, invoked the do: method rather than accessing the elements directly. This might have had unexpected side effects. (#36065)

Updates to persistent symbol list did not affect transient symbol list

Modifying your own UserProfile’s persistent symbol list should have updated the session’s transient symbol list. It did not. (#36274)

Blocked stone may unfairly terminate gems

If a stone is blocked, for example waiting for a response from a slow remote cache, gems may be terminated after the STN_GEM_TIMEOUT timeout when they should not be. (#36403).

Idle stone will not send lostOTRoot

If the Stone does not have any commits to process after a sigAbort is sent to a session, it will not send the sigLostOTRoot after the timeout. (#30027)

Duplicate reads into remote caches

Pages that are needed at the same time by multiple remote gems on the same cache were being read multiple times, by the pgsvrs of each remote gem. However, only the first gem to complete the page read into the remote cache could succeed; the work done by other pgsvrs and gems was discarded once the first gem's read was complete. Now, subsequent gems will wait for the first page gem to complete the read process. (#30900)

VSD display problems on Linux

Linux and AIX only

VSD running on Linux and AIX had multiple display problems. On Linux, most noticeably the y-axis labels were incorrect by an amount equivalent to one labeling increment. (#35143, #32971)

Daylight Savings Time starts and ends a second late

Daylight Savings Time officially occurs at specified time such as 2:00:00 AM, meaning that there should be no time 2:00:00 AM. The local time at the instant following 1:59:59 will be 3:00:00 AM. DateTime instances did not display the DST time until the second following, which becomes 3:00:01. (#36374)

Chapter 2

Extended Character Set Support

Past releases of GemStone have provided limited ability to process and collate String data that includes extended Characters. Earlier releases did not handle Characters over 127, more recently this limit was raised to 255. This is still insufficient for many languages.

With GemStone/S 6.2, support for extended character sets is added. This allows the various Character test, comparison, and conversion methods to work correctly across the full range of Characters that can be represented in one or two bytes. Users now have control over the contents of the character data tables that drive these methods, and can either use tables based on the Unicode Standard, or can design their own based on their particular application requirements.

Customers who do not require more than the basic 256 Character set do not need to do anything; GemStone by default uses basic 256 Character set tables, which provide the best memory space usage. Extended Character Set support was designed to provide the additional abilities without affecting applications that do not need them.

The new Extended Character Set support includes:

- Default Built-in Tables — by default, GemStone/S 6.2 comes with basic character set tables for the 0-255 Character range, based on the Unicode Standard.
- 16-bit Unicode Standard — users can extend the character data tables to support the Unicode Standard up through all Characters that can be represented with two bytes (0-65535).
- Flexible upgrade and customization — users can build their own character data tables, or update their applications when there are new releases of the Unicode Standard.

WARNING

The power to modify character tables means it is also possible to corrupt these tables in a way that breaks topaz command line processing. Use caution in using this new feature. If a corrupt character table is installed that renders the system unusable, see "To fix problems after installing an invalid character data table" on page 34.

Character Set Representation

Categories

Earlier releases included a Character type, which was represented as a number or a symbol. These types were #alpha, #digit, or #special. These types are deprecated and should no longer be used.

The replacement for this is Character categories. The categories are based on the Unicode standard, which provides categories for all possible Characters. Table 2.1 lists all the Unicode categories. While some of these Character categories are commonly used, other categories are more rare and not used in English or Latin based languages.

Titlecase category

In addition to uppercase and lowercase, the Unicode standard provides an additional case related category, Titlecase, category #Lt. This is a special case for composite characters that incorporate multiple individual characters. (Such characters are rare; there are only 12 such characters in the Unicode standard). An example is Unicode Character code 0x01C5 (decimal 453), Dž. Titlecase of these composite character forms consists of an Uppercase first character, followed by lowercase remaining characters. The uppercase would be DŽ, the lowercase would be dž, and the titlecase Dž.

For most letters, Titlecase is the same as Uppercase. GemStone's formatted tables may omit Titlecase.

Table 2.1 Character Category Codes and Symbols

1	`#Lu`	`Letter, Uppercase`
2	`#Ll`	`Letter, Lowercase`
3	`#Lt`	`Letter, Titlecase`
4	`#Lm`	`Letter, Modifier`
5	`#Lo`	`Letter, Other`
6	`#Mn`	`Mark, Nonspacing`
7	`#Mc`	`Mark, Spacing Combining`
8	`#Me`	`Mark, Enclosing`
9	`#Nd`	`Number, Decimal Digit`
10	`#Nl`	`Number, Letter`
11	`#No`	`Number, Other`
12	`#Pc`	`Punctuation, Connector`
13	`#Pd`	`Punctuation, Dash`
14	`#Ps`	`Punctuation, Open/Start`
15	`#Pe`	`Punctuation, Close/End`
16	`#Pi`	`Punctuation, Initial Quote`
17	`#Pf`	`Punctuation, Final Quote`
18	`#Po`	`Punctuation, Other`
19	`#Sm`	`Symbol, Math`
20	`#Sc`	`Symbol, Currency`
21	`#Sk`	`Symbol, Modifier`
22	`#So`	`Symbol, Other`
23	`#Zs`	`Separator, Space`
24	`#Zl`	`Separator, Line`
25	`#Zp`	`Separator, Paragraph`
26	`#Cc`	`Other, Control`
27	`#Cf`	`Other, Format`
28	`#Cs`	`Other, Surrogate`
29	`#Co`	`Other, Private Use`
30	`#Cn`	`Other, Not Assigned`

Configuring your application to use Extended Character Sets.

If your application uses more than the standard 256 Characters, you will need to define Character Data Tables that include the additional Characters, in addition to the basic 256. We recommend that you only add Characters that you require. The full Unicode allows for 65K Characters, but larger tables take up significantly more space.

GemStone represents Character data in two ways. For efficient loading during login, the tables are stored as a primitive character table, an Array of ByteArrays, each containing numerically encoded information. For users who need to view or edit the Character data, the same information can be represented as a structured table, the structured character table, consisting of an Array of Arrays with Character instances and Symbols.

Some methods work with structured table information, while others expect the lower-level ByteArray representation.

#CharacterDataTables

GemStone/S 6.2 adds a new variable in the Globals SymbolDictionary, #CharacterDataTables. This variable is used to hold a user-defined primitive character table. This variable is checked during login for each session; if not set, the session uses the default built-in 256 character tables. If the variable is set, the ByteArray tables stored in it are loaded into the internal tables and used for all subsequent character operations.

Primitive Character Table

A primitive character table consist of three tables:

a. the Dispatch Table, mapping unicode value to index in the main table.

b. the Main Table containing the numeric data values for uppercase and lowercase or numeric information for characters representing numbers. This table is in collation sequence order.

c. the TitleCase Table containing information for Characters with titlecase forms, if any.

Structured Character Table

To assist the user in viewing, updating, and modifying these tables, a more structured format is used for representing character data. A structured character table is formatted as an Array of elements, arranged according to character collate order. Each element is an Array of 4 or 5 entries:

a. The character for this entry.

b. The symbolic character category code.

c. Uppercase character (if a letter) / Numerator (if numeric).

d. Lowercase character (if a letter) / Denominator (if fraction).

e. (Optional) Titlecase character

For example, in the default structured character table the 73rd element is the letter $a (unicode/ascii 97). The array at this index contains the following:

```
#( $a #Ll $A $a )
```

The equivalent line in the printed output of the primitive character table would be:

```
65/65:  1  65  97
```

Corresponding to:

```
(table index)/(unicode) (category Code) (uppercase) (lowercase)
```

Loading Extended Character Sets

GemStone provides the 65K Unicode Character Data Set, which you can load into your repository to allow you to immediately work with all Characters.

To load this into your image, as SystemUser, execute the following and commit:

```
Character activateCharTablesFromFile:
         '$GEMSTONE/goodies/CharTableUnicode.dat'
```

For a customized version of the Character Set, you can load the primitive character table to a structured character table, and modify the structured character table by adding the new Characters you will be using. You can then export this to a file in passivated format, and load it into the primitive character table.

To do this:

1. Create a structured character table, and place this in a temporary variable. To begin with, you may wish to load the default 256 basic character table.

   For example:

   ```
   | passivatedData |
   passivatedData := PassiveObject fromServerTextFile:
      '$GEMSTONE/goodies/CharTableDefault.dat'.
   UserGlobals at: #MyCharacterDataTables put:
      (passivatedData activate)
   ```

2. Add the required entries to this table. Entries are Arrays formatted according to the description on page 2-28. They must be positioned in the table (Array of Arrays) in the desired sort order, not added to the end (unless the new Character should be sorted last in any collation sequence).

For example, if you would like to include the characters Š (Unicode 352) and š (Unicode 353), which would follow S and s in collation sequence respectively, you might do the following

```
MyCharacterDataTables
   add: (Array
         with: (Character withValue: 352)
         with: #Lu
         with: (Character withValue: 352)
         with: (Character withValue: 353))
   after: #( $S #Lu $S $s ).
```

```
MyCharacterDataTables
   add: (Array
         with: (Character withValue: 353)
         with: #Ll
         with: (Character withValue: 352)
         with: (Character withValue: 353))
   after: #( $s #Ll $S $s ).
```

3. When you are sure your changes are correct, as SystemUser, execute the following and commit:

   ```
   Character installCharTables: MyCharacterDataTables
   ```

   This converts the structured character table into equivalent byte arrays and places them into Globals variable #CharacterDataTables, to be loaded by all later sessions on login.

   Use caution: Note that if the table is incorrectly formatted, it is possible you may render your system unusable; see “To fix problems after installing an invalid character data table” on page 34. You may also affect or break indexes that rely on String sequencing. We recommend you drop all indexes in your System, if any, prior to modifying the character tables.

4. If you wish to make further additions based on the currently customized primitive character table, you may create the structured character table based on your current image information by executing:

   ```
   Character charTables
   ```

   This reads the primitive character table in #CharacterDataTables and returns the equivalent structured character table.

Image changes - Character

Many existing Character instance methods have been modified to use the new features of the Extended Character Set Support; for example:

```
isDigit
isLetter
isLowercase
isUppercase
isSeparator
isVowel
```

The following methods have been added:

`Character >> isTitlecase`
Returns true if this character is a Titlecase or Uppercase character, false otherwise.

`Character >> asTitlecase`
Returns the Titlecase version of this character, or the Uppercase if there is no Titlecase.

`Character Class >> categoryId:` *aSymbol*
Given a character category *aSymbol*, returns the numeric ID.

`Character Class >> categorySymbol:` *anInteger*
Given a character category ID *anInteger*, returns the category symbol.

`Character Class >> charTables`
Returns the structured representation for the character data tables recorded in Globals variable #CharacterDataTables.

`Character Class >> activateCharTablesFromFile:` *aFileName*
Installs the character data table information recorded in GemStone passivate format in the designated file to the Globals variable #CharacterDataTables. See the method `passivateCharTablesToFile:` for the method that generates this file.

`Character Class >> passivateCharTablesToFile:` *aFileName*
Writes the contents of the Globals variable #CharacterDataTables to the designated file using the GemStone passivate mechanism. See the method `activateCharTablesFromFile:` for the method that reads this file back into CharacterDataTables.

`Character Class >> installCharTables:` *tableInfo*
Converts a structured character data table into appropriately formatted byte arrays and then places them into Globals variable #CharacterDataTables for use in this and subsequent sessions. You must be SystemUser to execute this method.

WARNING: Installing incorrectly formatted character table data will break Character and String test, comparison, and conversion operations, including command line processing, to a point where the system will be impossible to use. If this occurs, see “To fix problems after installing an invalid character data table” on page 34.

Changing the collation order of Characters will break any indexes that are keyed off of Strings / DoubleByteStrings. Before changing the tables, remove all such indexes, install the new tables, and then reconstruct the indexes.

Files

There are two new files supplied in the $GEMSTONE / goodies directory. These files are in GemStone passivate format.

`CharTableDefault.dat`
Character table data for the default table, in structured form. This file can be used for viewing the contents of the default built-in tables.

`CharTableUnicode.dat`
Character table data for 16-bit Unicode, in low-level ByteArray form. This file is used to install the extended Unicode tables as described below.

The Unicode Database

The Unicode Consortium is an international standards organization that produces the Unicode Database, which provides unique codes for all Character in all Character Sets. The database continues to develop; GemStone's implementation is using version 4.1.0.

For more information on this database, refer to:

```
http://www.unicode.org/Public/UNIDATA/UCD.html
```

The Unicode Consortium provides code charts by script as well as a single master list of all characters, presented in an ASCII-only, comma delimited version. Version 4.1.0 of this file has been used to create the ByteArray tables, and is provided in passivated object form.

This table can be found at

```
http://www.unicode.org/Public/UNIDATA/UnicodeData.txt
```

If the standard file changes, and you need to recreate the internal tables, GemStone provides the following utility:

`$GEMSTONE/goodies/UnicodeData.gs`
The file UnicodeData.gs contains GemStone Smalltalk code that can be filed in as SystemUser to create the UnicodeData class, which includes various utilities for working with the Unicode Character Database.

Methods of interest in the UnicodeData class include:

`UnicodeData Class >> loadFromFile:` *fileName*
Load the contents of the Unicode Database from the specified file.

`UnicodeData >> generateTables`
Generate structured character data tables from the Unicode data. The result from this method can then be installed using `Character>>installCharTables:`.

Procedures

The following section describes several common procedures for using the new extended character set features of GemStone/S 6.2.

View current character data tables

To view the contents of the internal character data tables currently in use:

1. Login a linked topaz session.
2. Execute:

   ```
   Character _dumpCharTables
   ```

This writes output to stdout (standard output), that is, the linked topaz session output.

View structured version of current

To get the structured version of the character data tables:

When using the default built-in 256 character data tables:

```
(PassiveObject fromServerTextFile:
         '$GEMSTONE/goodies/CharTableDefault.dat') activate
```

When using Globals variable CharacterDataTables to override the default:

```
Character charTables
```

To install the 16-bit Unicode Standard character data table

1. As SystemUser, execute:

   ```
   Character activateCharTablesFromFile:
            '$GEMSTONE/goodies/CharTableUnicode.dat'
   ```

2. Commit

To install a specially tailored character data table

1. Construct your table as required, using formatting information provided earlier. Store someplace accessible, for example:

   ```
   UserGlobals at: #MyCharacterDataTables
   ```

2. As SystemUser, execute:

   ```
   Character installCharTables: MyCharacterDataTables
   ```

3. Commit

To install updated standard character data table

To install an updated 16-bit Unicode Standard character data table: (for when new releases of the Unicode Standard come out)

1. Download the updated Unicode Database at:

   ```
   http://www.unicode.org/Public/UNIDATA/UnicodeData.txt
   ```

   to a file (note that this URL may change). We'll call this file "unicode.dat"

2. Login a topaz session as SystemUser
3. File-in and commit $GEMSTONE/goodies/UnicodeData.gs
4. Execute the following:

   ```
   | tables |
   UnicodeData loadFromFile: 'unicode.dat'.
   tables := UnicodeData generateTables.
   Character installCharTables: tables.
   ```

5. Commit

To save the current #CharacterDataTable value to a file

To save the current contents of the Globals variable #CharacterDataTables to a passivated file, execute:

```
Character passivateCharTablesToFile:  'MyFileName'
```

To load a file into the current CharacterDataTables

To install the saved contents of a passivated file to the Globals variable #CharacterDataTables:

1. As SystemUser, execute:

   ```
   Character activateCharTablesFromFile:  'MyFileName'
   ```

2. Commit

To reset #CharacterDataTables to use the default 256 character set

1. Login as SystemUser
2. Execute:

   ```
   Globals removeKey: #CharacterDataTables.
   ```

3. Commit

To fix problems after installing an invalid character data table

1. At OS level, set the host environment variable GS_DISABLE_CHARACTER_TABLE_LOAD to TRUE (the particular value of this environment variable does not matter; its existence is the critical factor)
2. Login a new topaz session as SystemUser.
3. Execute Globals at: #CharacterDataTables put: nil.
4. Commit

Chapter

3

New TimeZone Subsystem

The rules covering the start and end of Daylight Savings Time (DST) in the US and Canada changed starting in March of 2007; Daylight Savings Time begins earlier, and ends later.

Instances of DateTime include a reference to a TimeZone object, which handles the conversion from the internally stored Greenwich Mean Time (GMT, also referred to as UTC or Coordinated Universal Time) and the local time. TimeZones encapsulate the DST rules, so a given GMT time is adjusted to local time for display. TimeZones are also used to calculate the internal stored GMT for newly created DateTime instances.

As a result, customers who had existing DateTime instances in US and Canadian time zones falling in the periods affected by the rules change may have had incorrect data. Affected customers should have loaded the TimeZone2007 patch and performed updates on existing DateTime instances as per the patch instructions.

A limitation of the old implementation of TimeZone is that only one rule was supported. This did not allow correct display localization of DateTime instances in the same TimeZone in different years in the case of rules changes. The new implementation of TimeZone addresses that problem, and the problem of keeping the rules up to date, by using the zoneinfo or tz public domain time zone database.

The new implementation of TimeZone is now part of the GemStone/S base, replacing the old implementation of TimeZone. All customers, including those who are not affected by US or Canadian TimeZones, will need to install default TimeZone instances following the new protocol.

Previously installed TimeZone2007 patch

If your application has already installed the TimeZone2007 patch, and ensured that existing instances of DateTime have appropriate time zone instances, you do not need to do anything further with this release and can disregard this chapter.

Setting the default TimeZone

Because the new TimeZone public API is the same as old TimeZone, there should not be any application programming changes required to use the new class. As in previous releases, there will continue to be a requirement that the customer DBA create and install a default TimeZone instance or instances appropriate for the customer's location and the application.

In addition to creating and installing the correct TimeZone instance, applications may need to update existing references to instances of the old TimeZone class, to reference instances of the new TimeZone class. Applications may also need to correct existing invalid DateTime instances.

Future rule changes for any TimeZone will require that each of these steps be repeated. The nature of the future rule change will determine the nature of any changes that need to be made to internal DateTime values.

installtimezone.txt

If this is a new application, or if you do not have existing instances of DateTime and TimeZone, you may use the script $GEMSTONE/upgrade/`installtimezone.txt` to install the correct TimeZone. This script can be edited using any text editor and input using topaz. The script comments include instructions on editing.

timezone.txt

If you have not previously installed the TimeZone2007 patch, a script, $GEMSTONE/upgrade/`timezone.txt`, is provided with more options for application update. This script can also be edited using any text editor and input using topaz. The `timezone.txt` script illustrates setting the default TimeZone instance, updating references to TimeZones, and updating incorrect DateTimes.

The requirements for updating TimeZone and DateTime instances are described in detail in the next section, **’Application update process’**.

Application update process

It is necessary for all customer DBAs to evaluate each of their applications, to determine if the change in DST rules has an impact on that application, and if so, what is the correct action to take. Customer applications may be impacted according to one of the following situations:

A. The application has no committed instances of DateTime, and does not use TimeZone directly. You may still update the default TimeZone instance, but no other action is needed.

B. The application has committed instances of DateTime that describe points in the past, or in the future that are not in the “redefined periods”; or the DateTime instances do not reference a time zone whose rules have changed.

   DateTime instances that are not in the redefined periods or not affected by the change in rules may continue to reference the old TimeZone instance. The new TimeZone must be installed and instances of this will be used for new instances of DateTime.

   The new implementation of TimeZone acquires time zone rules automatically from an standardized universal database. If you are in a non-US time zone, or you have modified GemStone to handle your TimeZone rules, it is important that you review the old and new rules to avoid unexpected changes in DateTimes in your application.

C. The application has committed instances of DateTime that describe points during the redefined periods; but the internal stored GMT is correct.

   When an instance of DateTime is created based on local time, the current default TimeZone is used to compute the stored value, which is in GMT. Since the rules were not correct for the redefined periods, the computation incorrectly computed the internal GMT times.

   However, there are DateTime instance creation methods that explicitly specify the GMT (DateTime `fromStringGMT:`, etc.). It requires application knowledge to determine if the internal GMT values may be correct for existing DateTimes that are in the redefined periods.

   If DateTime internal GMT times are correct, the instances of DateTime should be modified to reference the new TimeZone, but the DateTime instances do not need to have their internal data updated.

D. The application has committed instances of DateTime that describe points during the redefined periods, and these DateTime instances have an incorrectly computed internal stored GMT.

   In this case, instances of DateTime should be modified to reference the new TimeZone, and will also need to have their internal GMT time corrected.

Table 3.1 describes the specific steps that will be required, depending on which of the above situations is true for your application. Following this table, the individual steps are described in more detail. GemStone provides a topaz script, **`timezone.txt`**, that provides code that will perform some of these steps automatically.

Important note on TimeZone references

The information provided in this documentation, and the scripts provided with GemStone, assume that the only use of TimeZone is by instance of DateTime. If your application has other objects that reference or rely on TimeZone - for example, an application-specific TimeStamp class - you **must** review your code and TimeZone usage to determine what changes, if any, are needed to the committed instances in your repository.

Table 3.1 Operations for Different Application Scenarios

	A, B **Applications that use DateTime or TimeZone**	**C** **Applications with existing instances of DateTime**	**D** **Applications with instances of DateTimes in modified periods**
(1) Find references to TimeZone instances			X
(2) Find existing TimeZone instances		X	X
(3) Create new TimeZone instances	X	X	X
(4) Install new TimeZone instances	X	X	X
(5) Map old to new TimeZone instances		X	X
(6) Swap old / new TimeZone instances		X	X
(7) Update objects referencing TimeZone instances			X

(1) Find references to TimeZone instances. If there are DateTimes that were created during the redefined periods, which have incorrect internal GMT times that will need to be adjusted, then instances of DateTime - references to the TimeZone instance(s) - need to be located. If there is no application specific way, then you may use:

```
    DateTime allInstances
```

This step may be combined with the following step, using `listInstances:`.

The timezone.txt script has the ability to collect and adjust DateTime values, however by default this is not done. If you wish to find and adjust all instances of DateTime, set the script variable **`move`** to true.

If you have both DateTimes that need adjustment, and DateTimes that do not need adjustment, or if you have DateTimes in more than one TimeZone that need adjusting, you will have to modify the script.

(2) Find existing TimeZone instances. If the old TimeZone instances need to be replaced, then the DBA will need to locate all TimeZone instances. In many cases, the TimeZone instances referenced by the class variables Default and TimeZones will be enough. If other instances of TimeZone have been created and referenced, then you may wish to use

```
    TimeZone allInstances
```

This step may be combined with the previous step, using `listInstances:`.

The timezone.txt script only updates the current default TimeZone. If you have more than one TimeZone in your application, you will need to modify the script.

(3) Create new TimeZone instances. There are a number of ways to do this, using TimeZone class instance creation methods. In general, you need to identify the path to a binary file that describes the time zone rules.

The timezone.txt script provides the following examples of creating the TimeZone instance:

- **If your application is on Solaris or Linux**, the operating system can provide a new TimeZone instance based on the current machine configuration using TimeZone Class >> fromOS.

  If this is the case, set the script variable **`USA4auto`** to false.

- **If your application default current TimeZone is in one of the four main USA time zones** (Eastern, Central, Mountain, or Pacific), GemStone can automatically create a new TimeZone instance using the time zone database provided with GemStone.

  If this is the case, leave the script variable **`USA4auto`** at the default setting of true.

- **If you wish to use GemStone's provided time zone database**, you may use the interactive script `$GEMSTONE/pub/timezone/etc/tzselect`. This script will ask a set of questions, and use the answers to provide the correct time zone descriptor name. With this, you can create the new TimeZone instance using the time zone database provided with GemStone.

  In this case, set the script variable **`USA4auto`** to false, and set the script variable **`path`** to the time zone descriptor name, for example:

  ```
  path := 'Europe/Zurich'.
  ```

- **If you wish to use your own time zone database**, and you know the time zone descriptor name for your TimeZone, you can specify the full path to the time zone information.

  In this case, set the script variable **`USA4auto`** to false, and set the script variable **`path`** to the full path to your time zone database, including the time zone descriptor name, for example:

  ```
  path := '$GEMSTONE/pub/timezone/etc/zoneinfo/Europe/Zurich'.
  ```

(4) Install new TimeZone instances. TimeZone class includes class variables that hold the current and default instances of TimeZone, and cache frequently used TimeZone instances keyed by name.

The timezone.txt script automatically performs the installation of the new default TimeZone. If there are other TimeZone instances used, you will need to modify the script to install them.

(5) Map old to new TimeZone instances. Each old TimeZone instance will require an instance of the new TimeZone. You must keep track of the mappings between the old and new TimeZone instances so each reference to a TimeZone will be updated to the correct new TimeZone.

If the timezone.txt script variable **`move`** is true, the mapping is created from the current default old TimeZone instance to the equivalent new TimeZone instance. If you have more than one TimeZone mapping, you will need to modify the script.

(6) Swap old/new TimeZone instances. In general, this will be handled with `become:`. Using the previously created map between old and new TimeZone instances, each old TimeZone instance will become the equivalent new TimeZone instance.

The timezone.txt script by default swaps current default TimeZone. If you do not want the current default TimeZone to be swapped, set the script variable **`swap`** to false. If you have more than one TimeZone in your application, you will need to modify the script.

(7) Update objects referencing TimeZone instances. If there are DateTime instances that were created during the redefined periods, which have incorrect internal GMT times, each of these DateTime instances needs to be adjusted. The old / new map from step 5 is used to determine the adjustment needed.

By default, the timezone.txt script does not update DateTime instances. If you wish to find and adjust references to all DateTime instances, set the script variable **`move`** to true.

If you have both DateTimes that need adjustment, and DateTimes that do not need adjustment, or if you have DateTimes in more than one TimeZone that need adjusting, you will have to modify the script.

Implementation

The widely used public-domain time zone database, ZoneInfo or tz, contains code and data that records time zone information for locations worldwide. It is updated periodically when boundaries or rules change in any of the represented locations.

Each record in the tz database represents a location where all clocks are kept on the same time as each other throughout the year, coordinating any time adjustments such as DST, and have done so for many years. Locations are identified by continent (or ocean, for islands) and name, which is usually the largest city within the region. For example, America / Los_Angeles, Europe / London, etc.

tz is provided initially as text files, which may be compiled into binary files using tz's compilers. GemStone's TimeZone implementation uses the compiled binary form, which is also used by the Solaris and Linux operating systems. To get updated source files, download from

```
ftp://elsie.nci.nih.gov/pub/tzdata*.tar.gz
```

The timezone sources in this file may be compiled using the zic timezone compiler, which GemStone provides (see "zic" on page 42).

New GemStone Classes

TimeZoneInfo - represents a geo-political region that shares time zone information, such as "America / Los_Angeles". The class side contains public methods for reading the ZoneInfo binary file for a particular zone. The instance side contains public methods to query for the UTC offset on particular dates and times. When the rules change, one can read the updated file and start using the new instance.

TimeZoneTransition - an internal class that contains some information about the moment of a transition from one offset to another offset.

Other ANSI compliant classes have also been added as part of the new subsystem: **DateAndTime**, **DateAndTimeANSI**, and **Duration**.

Updated **TimeZone** class - a new version of the class TimeZone, with an API very similar to the current (old) implementation of this class. It is intended that customer code not require changes to work with the new TimeZone class, although any modification to the old TimeZone class or calls to private methods in TimeZone are likely to require modification.

Utilities

tzselect, **zdump** and **zic** are public domain, open source utilities that are useful in working with the zoneinfo database. These utilities are provided with the Solaris and Linux operating systems. For the convenience of users on HP-UX and AIX, GemStone provides these utilities along with the other zoneinfo database files.

Note that these are not GemStone utilities, and they are not supported by GemStone.

You may download the source code for these utilities here:

```
ftp://elsie.nci.nih.gov/pub/tzcode*.tar.gz
```

Documentation for these utilities is provided as man pages. To read the man pages, add the directory $GEMSTONE/pub/timezone/man to the MANPATH.

To run these, you may wish to add $GEMSTONE/pub/timezone/etc to the executable path.

tzselect

tzselect allows to you interactively select a time zone. The interactive script asks you a series of questions about the current location and outputs the resulting time zone description to standard output. The output is suitable as a value for the TZ environment variable and GemStone scripts.

You may need to set the environment variable $TZDIR to $GEMSTONE/pub/timezone/etc/zoneinfo (or the path to your zoneinfo database, for this script to work correctly. You may also need to set the environment variable $AWK, to any POSIX compliant awk program.

For further details on using tzselect see the man page.

zdump

```
zdump [-v] [-c cutoffyear] [zonename...]
```

zdump prints time zone information. It prints the current time for each time zone (zonename) listed on the command line.

Specifying an invalid zone name to zdump does NOT return an error; instead, it returns the zdump output for GMT. This reflects the same behavior of the time routines in libc.

The -v option will display the entire contents of the time zone database for the given time zone name.

For further details on using zdump, including the command line options, see the man page.

zic

```
zic [-s] [-v] [-l localtime] [-p posixrules]  [-d directory]
[-y yearistype] [filename...]
```

zic compiles time zone source files. It reads input text in files named on the command line, and creates the time zone binary files.

To create files in a specific location, rather than the standard platform directory (on Solaris, /usr/share/lib/zoneinfo), use the -d *directory* option.

For example, to recompile sources on Solaris to the GemStone timezone database, execute the following:

```
zic -d $GEMSTONE/pub/timezone/etc/zoneinfo/
/usr/share/lib/zoneinfo/src/northamerica
```

For further details on using zic, including the command line options and the structure of the source code files, see the man page for zic.