GemStone/S 64 Bit™
Installation Guide

for MacOS on Intel/AMD
or Apple silicon
(Development only)

Version 3.7.2

December 2024

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.

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 has been 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".

TRADEMARKS

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

UNIX is a registered trademark of The Open Group.

Intel is a registered trademarks of Intel Corporation.

Microsoft, **Windows**, **Windows Server**, and **Azure** are registered trademarks of Microsoft Corporation.

Linux is a registered trademark of Linus Torvalds and others.
Red Hat, **Red Hat Enterprise Linux**, **RHEL**, and **CentOS** are trademarks or registered trademarks of Red Hat, Inc.
Rocky Linux is a trademark or registered trademark of Rocky Enterprise Software Foundation.
Ubuntu is a registered trademark of Canonical Ltd., Inc.

AIX, **Power**, **POWER**, **Power8**, **Power9**, and **VisualAge** are trademarks or registered trademarks of International Business Machines Corporation.

Apple, **Mac**, m**acOS**, and **Macintosh** are trademarks of Apple Inc.

Instantiations is a registered trademarks of Instantiations, Inc.

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

Raspberry Pi is a trademark of the Raspberry Pi Foundation.
RabbitMQ is a trademark of VMware, Inc.
Prometheus is a registered trademark of The Linux Foundation.
Grafana is a registered trademark of Raintank, Inc. dba Grafana Labs.

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

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

Preface

About This Documentation

This document explains how to install GemStone/S 64 Bit version 3.7.2 on a workstation running MacOS on Intel/AMD x86_64 or on Apple silicon (arm64); and how to upgrade from previous GemStone/S 64 Bit versions.

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 and HTML.
- **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.
- **Supplemental Documentation** and **TechTips**, providing information and instructions that are not in the regular 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 4. Upgrading GLASS/GsDevKit Applications

Chapter 5. Configuring GBS for GemStone/S 64 Bit

Chapter

1 Installing GemStone/S 64 Bit Version 3.7.2

This chapter describes the procedure for installing GemStone/S 64 Bit™ version 3.7.2, and for additional OS configurations that can optimize your GemStone application.

If you are upgrading to this release from a previous version of GemStone/S 64 Bit, after installing v3.7.2, follow the instructions in the appropriate later chapter of this Installation Guide. These upgrade instructions will provide details on steps that need to be taken before and after the installation described here.

GemStone/S 64 Bit on MacOS is supported for development use only, not for use in production.

Review the Installation Procedure

The following list summarizes the steps to install GemStone/S 64 Bit.

Check the System Requirements

Before you install GemStone/S 64 Bit, ensure that the following system requirements are satisfied. Additional system resources may be necessary to support large applications.

Platform

- Intel/AMD x86_64
- Apple silicon (ARM)

RAM and Swap space

- While small installations can run on systems with only a few GB of physical RAM, increasing RAM, and so allowing a larger Shared Page Cache, is important for GemStone performance.
- Adequate free disk space on the machine's boot partition beyond other system needs.

Disk space

- Space for the installed distribution files—you need approximately 800MB for GemStone/S 64 Bit, and additional space for other products.
- Additional space as required for your repository.

Operating system

- MacOS 14.5 (Sonoma) with Darwin 23.5.0 kernel on Apple silicon (ARM)
- MacOS 14.5 (Sonoma) with Darwin 23.3.0 kernel on x86_64
- MacOS 11.7.10 (Big Sur) with Darwin 20.6.0 kernel on Apple silicon (ARM)
- MacOS 11.7.10 (Big Sur) with Darwin 20.6.0 kernel on x86_64

Debugger

A C debugger allows C-level stack traces when a GemStone error occurs, or when using the **pstack** command. While not required for GemStone execution, it is strongly recommended that a debugger be installed, so diagnostic error stacks will be available.

- lldb 12.3.0

  lldb is installed as part of the Xcode command line tools package.

  You must be logged in as root, or as a user with administrative privileges, in order to use a debugger or to generate stack traces from GemStone processes.

C/C++ Compiler

GemStone requires a C/C++ compiler if you are parsing C header files for creation of FFI interface classes, as described in the *Programming Guide*, or developing C or C++ code for user actions or a C or C++ application; as described in the *GemBuilder for C* manual.

If you have committed your FFI generated code, or compiled your user actions or application, the C/C++ compiler is not needed for application execution.

The compiler is required for application execution only when using GsTsExternalSession to access remote Stones running a different version of the GemStone server.

- Sonoma: Apple clang version 15.0.0 (clang-1500.3.9.4)
- Big Sur: Apple clang version 13.0.0 (clang-1300.0.29.30)

X Windows

An X Windows server allows you to use GemStone's graphical VSD application on Mac. Alternatively, GemStone statistical data may be viewed on Windows, by transferring the data files, or by mounting the file system on Windows. X Windows is not required for GemStone execution.

For further OS configuration and optimizations, see "Additional Configuration of the Operating System" on page 15

Install GemStone v3.7.2 distribution

The GemStone distribution should be installed as a regular user, not as the root user. Some later installation steps do require login as root. Starting GemStone and and GemStone administration should be done as a regular user, not as root.

For GemStone administration, select an existing user, or create a specific user to be the GemStone administrative user. This administrative user is referred to in these instructions as *gsAdminUser*.

1. Log in as *gsAdminUser* (the GemStone administrator) to the machine on which you are installing GemStone. This part of the installation should **not** be done as root, to ensure all the files are not owned by root.
2. Determine that adequate swap space is available.:
3. Select the drive on which you will install the GemStone software, and the installation directory on this drive, *InstallDir*. Make this directory the current working directory.

   We recommend that you avoid choosing either an NFS-mounted partition or one containing OS swap space for the initial installation. Mounted partitions can result in executables running on the wrong machine and in file permission problems. Existence of swap space on the same drive can dramatically slow GemStone disk accesses.
4. GemStone/S 64 Bit is provided as a .dmg file with a name similar to: `GemStone64Bit3.7.2-i386.Darwin.dmg` (on Intel/AMD), or `GemStone64Bit3.7.2-arm64.Darwin.dmg` (on Apple silicon).

   NOTE

   While Intel/AMD executables can run on Apple silicon, it would run in emulation mode; performance may be slow. You should download the correct distribution for your system.
5. Double click on this file to open it, and drag the installation tree to *InstallDir*.
6. The *InstallDir* now contains a GemStone directory with a name similar to `GemStone64Bit3.7.2-`*platform*`.Darwin`.

The GemStone distribution is now installed.

In addition to subdirectories, the installed GemStone distribution directory also contains the text files `version.txt`, which identifies this particular product and release of GemStone, and `externals.sha.txt`, which lists SHAs for external projects included in this version.

Configure the GemStone Installation

After installing the GemStone distribution, you need to set or reset environment variables, ensure you have the appropriate keyfile installed, and that your NetLDI user is configured correctly.

Set Environment Variables

To use GemStone, you will need to define the GEMSTONE environment variable, and in most cases, set the OS search path.

1. The environment variable GEMSTONE should be set to the full pathname (starting with a slash) of your new GemStone installation directory. For example:

   ```
   os$ export GEMSTONE=InstallDir/GemStone64Bit3.7.2-platform
   ```

2. Set your OS search path to include the $GEMSTONE/bin directory.

   ```
   os$ export PATH=$GEMSTONE/bin:$PATH
   ```

3. If this machine has run previous versions of GemStone, or will run multiple versions, it is important to review all GemStone-related environment variables for ones that are incorrect for the new version, such as GEMSTONE_NRS_ALL, GEMSTONE_EXE_CONF, and GEMSTONE_SYS_CONF.

   Check for existing GemStone environment variables:

   ```
   os$ env | grep GEM
   ```

If any environment variables exist that are not appropriate for the new installation, you must specifically unset each one.

Set the GemStone Keyfile

The GemStone/S distribution includes a community keyfile, which will be used if a keyfile is not specified.

You may use a licenced customer keyfile (that is, a GemTalk-provided keyfile for your license) of the correct platform from any v3.7.x version with v3.7.2; keyfiles from 3.6.x and earlier are not valid with v3.7.2.

A keyfile must be located where GemStone can find it on startup, in order:

- A file specified by the KEYFILE configuration parameter in the configuration file used by the stone. This is not set by default, but may be defined to read a keyfile with any name in any location.
- `$GEMSTONE/sys/gemstone.key`
- `$GEMSTONE/sys/community.starter.key`

Licensed Customer keyfile

You may continue to use your v3.7.x keyfile with version 3.7.2. If you are upgrading from v3.6.x or earlier and need a new keyfile, or you have questions about your keyfile or license limits, email keyfiles@gemtalksystems.com, or contact GemTalk Technical Support.

Community keyfile

The GemStone distribution includes a community keyfile, `community.starter.key`, with product and system limits per the Community and Web Edition License. See https://gemtalksystems.com/licensing for details on the license terms.

Installing a keyfile

The **configuregs** and **installgs** scripts offer to install a keyfile for you.

To specify the location and name of a customized keyfile using the KEYFILE configuration parameter, edit the configuration file that will be used by the v3.7.2 stone to include the location and name of the keyfile.

You may also put the keyfile in the default location, `$GEMSTONE/sys/gemstone.key`. This requires adding write permission to the `$GEMSTONE/sys` directory; ensure that you remove write permission after the update.

Configure Services for a Named NetLDI

The NetLDI permits remote processes to interact with the repository. A NetLDI is required for some local and all remote sessions to log into GemStone. Logins can specify a NetLDI by name or by port number.

Defining a NetLDI service name is only necessary if you will use named NetLDIs. If you start a NetLDI using a port number, then no configuration of the system services database is needed. The login parameters can use the NetLDI's port rather than a name.

If you are using named NetLDIs, and your configuration includes multiple machines, (including clients on Windows or other platforms), you must update the services database on each node with an entry mapping the NetLDI name to the same port number.

To configure a named NetLDI:

1. Determine how the services file is setup on your system. If your system is using /etc/services, the **installgs** script can setup a named NetLDI for you.Otherwise, consult your system administrators.
2. Determine whether a service for the NetLDI name, such as `gs64ldi`, is already defined. The GemStone distribution includes an executable that will allow you to check for a definition:

   ```
   os$ $GEMSTONE/install/getservbyname gs64ldi
   s_name=gs64ldi s_port = 50377 s_proto = tcp
   ```

3. If your desired NetLDI name is not defined, add an entry similar to the following to the system services database:

   ```
   gs64ldi 50377/tcp #GemStone/S 64 Bit 3.7.2
   ```

   Choose a port number that is not being used by another service. The port number should be in the range 49152 <= port <= 65535, to conform to IANA standards (http://www.iana.org/assignments/port-numbers).

If several machines will be running GemStone, the system services database must be updated for each machine, including Windows clients as well as UNIX nodes. The port number must be the same for every machine.

Repository configuration using configuregs

If you are installing GemStone on a machine that has not previously had a version of GemStone installed, running **configuregs** (or the equivalent manual process) is required to perform the setup of the `/opt/gemstone/locks` and `log` directories. These directories must exist and be accessible in order to run GemStone.

If the `/opt/gemstone/locks` and `log` directories already exist, running **configuregs** is optional.

The **configuregs** script must be executed with root permissions; either logged in as root, or using sudo. the $GEMSTONE environment variable must be defined.

```
os$ sudo -E $GEMSTONE/install/configuregs
```

This has the following options:

- `Enter the user that will own GEMSTONE (required, must exist and not be root)`

  The files in the installation will be chowned to be owned by this user. Enter *gsAdminUser*.

- `Enter the group that will own GEMSTONE (required, must exist):`

  The files in the installation will be chgrouped to this group. Depending on your NetLDI configuration, all users that will use GemStone should be in this group.

- `Enter a key file to use with GEMSTONE (optional):`

  Enter the path to a keyfile, if you have a keyfile you wish to install. If you leave this empty (the default), by pressing enter, the Stone may be started using the community key file, or you may install your own key file later. If you supply the path to a keyfile, this is renamed and copied, to $GEMSTONE/sys/gemstone.key.

- `Should the netldi and port number be added to /etc/services ?: ['n']`

  configuregs can add the NetLDI name and port number to /etc/services. Unless you are familiar with this step and know your port lookup goes through /etc/services, use the default `n`.

- `Protect GemStone processes from the Linux Out of Memory killer?: ['y']`

  configuregs can add Out-of-Memory killer protection to key GemStone process. If you answer the default `y`, **configuregs** will add the **`cap_sys_resource`** capability to certain executables.

- `Update /etc/systemd/logind.conf to prevent killing GemStone at logout?: ['y']`

  On some Linux distributions, the default for **RemoveIPC** is yes, which can result in the GemStone repository dying with an error when the user that started GemStone logs out. To be safe, we recommend explicitly setting RemoveIPC=no in /etc/systemd/logind.conf. Answering yes to this question will add the line to this file, unless it is already present.

Reusing the configuration instructions

configuregs writes a file `answers.txt` in the $GEMSTONE/data directory. This is a plain-text that can be edited if desired, and used to repeat the operations done by

configuregs without re-answering each question, if you are installing on a different host or on a new GemStone release.

```
os$ sudo -E $GEMSTONE/install/doconfiguregs
   /var/tmp/gsanswers.txt
```

After **configuregs** completes, the GemStone distribution is ready to use.

- If you are new to GemStone or setting up a new repository, continue with "Additional Configuration of the Operating System" on page 15.
- If you are upgrading from an older version, continue with the upgrade process following the step on page 26.
- If you are familiar with your application requirements and setting up on a new machine, see "Additional Configuration of the Operating System" on page 15.

Repository configuration using legacy installgs

The legacy **installgs** script verifies your environment, creates lock and log file directories, sets up the extent files, and can be used to configure certain types of security for multi-user systems.

For new users and for test and development systems without strict security requirements, it is recommended to use the **configuregs script (**on page 12), and not running **installgs**.

Running **installgs** is needed if you want to configure the NetLDI to run in root mode with the s bit set.

The **installgs** script must be executed with root permissions; either logged in as root, or using sudo. the $GEMSTONE environment variable must be defined.

```
os$ cd $GEMSTONE/install
os$ sudo ./installgs
```

This has the following options:

- Set up directories for server lock files and NetLDI logs?

  The default location for server lock files and NetLDI log files is `/opt/gemstone`. If the directory does not exist, the installation script asks a number of questions allowing you to create `/opt/gemstone` and the subdirectories `locks` and `log`, and to set access (770) to these directories.

- Set the owner and group for all the files in the distribution?

  If you answer **yes**, the script will prompt you for the owner and group you want to use. Refer to Chapter 1 of the *System Administration Guide* for more information about setting owner and group permissions.

  If you answer **no**, the permissions will remain the same as when the files were extracted from the distribution media.

- Protect the repository extent file?

  The default gives only the owner read and write access (600) through ordinary UNIX commands. Other users can read and write the repository extent through a GemStone session. If you choose not to protect the repository, the executables run under ownership of the user who invokes them.

  Default: Set the repository permission to 600, and leave the setuid bit applied.

- Allow NetLDI to Run as Root?

  The NetLDI permits remote processes to interact with the repository. There are two ways to set up a NetLDI so that it can provide services to all GemStone users: it can run as root, or it can run in guest mode with a captive account.

  To run NetLDIs as root, accept the default "yes" response. Ownership of the NetLDI executable is changed to root, and the setuid bit is set. Any GemStone user will be able to start a NetLDI process that is accessible to all GemStone users because it will always run as root. For certain services, users will need to authenticate themselves by supplying a password.

  To run NetLDIs in guest mode with a captive account, answer "no" to the prompt, because those modes are not permitted if the NetLDI runs as root. "Guest mode" means that GemStone users do not have to supply a UNIX password to use NetLDI services. The "captive account" is an account that owns all processes the NetLDI starts; typically, it is the GemStone administrative account that owns the files. You must start the NetLDI while logged in as that account.

  If you are running a development system without strict security requirements, we recommend not using the default.

  Default: Change ownership of the `netldi` executable to root, and set its setuid bit.

- Set up an Extent?

  installgs offers to set up a new GemStone repository in $GEMSTONE/data. This is only appropriate if you wish to create a new repository. It performs the equivalent of the setup described in "Simple classic repository setup in data directory" on page 20.

  Default: Place a writable copy of `extent0.dbf` in `$GEMSTONE/data`.

- Start a NetLDI?

  installgs offers to start a NetLDI that is running in root with s-bit set; this should only be used if you have answered yes to "Allow NetLDI to Run as Root?".

  Default: Do not start a NetLDI at this time.

After running installgs, log out as user root. Further work is done as the GemStone administrative user.

GemStone/S 64 Bit v3.7.2 is now installed and configured.

To create a test repository, see Chapter 2, "Setting up a new Repository".

If you are upgrading from an earlier version, continue with the instructions in Chapter 3.

Additional Configuration of the Operating System

There are a number of Operating System-level configurations that can affect your GemStone environment. This includes a number of settings that are important for very large applications, but are not needed for simple test installations.

The kernel must be configured to support shared memory and semaphores. See your operating system documentation for further information. These requirements apply both to server nodes and to client nodes.

1. Kernel parameters

To specify shared memory and other kernel parameters, you must create a plist file with the name `com.gemtalksystems.shared-memory.plist`, and put this file in `/Library/LaunchDaemons/`. See below for an example.

Shared Memory

The maximum shared memory segment should be set to a value larger than your desired Shared Page Cache size, and not more than 75% of your real memory size. In addition, for most systems you will need to increase the system-wide limit on shared memory segments.

For example, if you have 8192 MB of real memory:

```
8192 MB * .75 = 6144 MB
6144 MB * 2^20 = 6442450944 bytes
```

The following settings would be defined:

```
kern.sysv.shmmax=6442450944
kern.sysv.shmall=1572864
```

Note that `kern.sysv.shmmax` is configured in bytes, but `kern.sysv.shmall` is configured in pages with a base page size of 4096.

Number of Processes

The default limits on the number of processes are adequate for up to about 250 GemStone users, for configurations establishing security via captive account. To allow more processes for the GemStone administrative userid, increase the settings for **`kern.maxprocperuid`** and **`kern.maxproc`**.

Setting kernel parameters

The following example contents of `com.gemtalksystems.shared-memory.plist` would apply to a system with 8MB of RAM, as described above.

You may download this file from https://docs.gemtalksystems.com/current/com.gemtalksystems.shared-memory.plist

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
  <key>Label</key>
  <string>shmemsetup</string>
  <key>UserName</key>
  <string>root</string>
  <key>GroupName</key>
  <string>wheel</string>
  <key>ProgramArguments</key>
  <array>
    <string>/usr/sbin/sysctl</string>
    <string>kern.sysv.shmmax=6442450944</string>
    <string>kern.sysv.shmall=1572864</string>
  </array>
  <key>KeepAlive</key>
  <false/>
  <key>RunAtLoad</key>
  <true/>
</dict>
</plist>
```

After defining the file or making changes to the settings, you will need to reboot so the changes take effect.

NOTE: On some version of macOS, you may need to run plist manually after reboot in order for the settings to take effect.

2. PAM

If you are using UNIX authentication for GemStone logins, or if you run NetLDI as root with setuid (i.e. not in guest mode), you must have PAM (Pluggable Authentication Module) configured on the server. You may include a specific GemStone authorization service name, or allow the default "other" authentication definitions to be used.

PAM authentication definitions are files under the directory `/etc/pam.conf/`.

The specific GemStone service file names are `gemstone.gem` for user authentication, and `gemstone.netldi` for a NetLDI running with authentication.

The libraries that are specified in the stack depend on how you are configuring PAM to perform the authentication. The examples below are for PAM configured to invoke LDAP for authentication.

For example, to allow GemStone UNIX authentication, which uses PAM, to authenticate via LDAP, create a file named `/etc/pamd.conf/gemstone.gem` with the following contents:

```
auth        required          pam_opendirectory.so
```

For NetLDI authentication, again using LDAP, create a file named `/etc/pamd.conf/gemstone.netldi` with the following contents:

```
auth        required          pam_opendirectory.so
```

You can allow the "other" authentication stack to be used for GemStone authentication by editing the file `/etc/pamd.conf/other` so it includes the following:

```
auth        required          pam_opendirectory.so
```

Consult your System Administrators for more information on how authentication is handled on your system.

3. System clock

The system clock must be set to the correct time. When GemStone opens the repository at startup, it compares the current system time with the recorded checkpoint times; a system time earlier than the last checkpoint time may be taken as an indication of corrupted data and prevent GemStone from starting. The time comparisons use GMT.

Chapter

2 Setting up a new Repository

This chapter describes how to setup a new GemStone repository.

For new users, we recommend that you create a GemStone repository initially on a single machine, to ensure that all the pieces work together. Chapter 1 of the *System Administration Guide* provides an introduction to GemStone, and this manual provides details on setting up configurations to meet specific application requirements.

Set up a new Repository

There are many ways to set up a GemStone repository, depending on your system requirements.

The repository setup instructions in this Chapter describes setting up a basic repository, suitable for a single user, or a small system with low security requirements.

Multi-user systems and system with stringent security requirements will need special configuration to allow multiple users to login and access the database while ensuring unauthorized users cannot access or modify data. Large or high-volume repositories require separation of the extents holding data, the transaction logs holding the records of changes, and the log files.

The options for setting up a GemStone repository that is optimized for larger, higher-volume, highly trackable and maintainable, and for more highly secure repositories are described in the *System Administration Guide*. Please consult GemTalk Technical Support for additional assistance.

The historic **installgs** utility, described under "Repository configuration using legacy installgs" on page 13, performs both configuration and also offers to setup a new repository. While this script is valuable for some securityconfigurations, it is not a recommended way to setup a repository.

Simple Data-conf repository setup using createNewGemStoneRepository

A Data-conf installation is a repository configuration in which all the repository-specific files are located in a single directory (one that is not within the GemStone distribution directory), and a specific configuration file, `gemstone_data.conf`, is used as the **-E** argument to GemStone processes.

While convenient for new users and small examples, this is not a recommended configuration for enterprise-level GemStone production environments. The *System Administration Guide* provides much more guidance on setting up a scalable, multi-user GemStone configuration.

The special configuration file `gemstone_data_conf`, makes setup of this environment easy. The Stone, NetLDI, and linked topaz startup commands use **-E** to specify this configuration file.

To set up a Data-conf repository, use the script **createNewGemStoneRepository**, passing in an argument of the path to your desired repository root directory. This directory must not exist or be empty, and the path must be writable.

For example, to create a new repository directory under the existing directory `/gshost/`, with the directory name `testGS`, do the following:

1. Create the repository using the script:

   ```
   os$ $GEMSTONE/install/createNewGemStoneRepository
      /gshost/testGS
   ```

   This will printout commands to start the Stone and NetLDI.

2. Start the stone, passing in the configuration file using the **-E** argument.

   ```
   os$ startstone -E /gshost/testGS/gemstone_data.conf
   ```

   This will start the stone with the default name **gs64stone**.

3. Start the NetLDI. This example starts the NetLDI in guest mode with captive account.

   ```
   os$ startnetldi -g -a gsAdminUser
   ```

   or

   ```
   os$ startnetldi -E /gshost/testGS/gemstone_data.conf -g -a
      gsAdminUser
   ```

   This will start the NetLDI with the default name **gs64ldi**. The **-E** argument to **startnetldi** is optional; it is only useful if you have other configuration file customizations that apply to Gem sessions.

Simple classic repository setup in data directory

The classic GemStone Installation uses the $GEMSTONE/data directory for all repository-specific files. This includes the extent file and the system configuration file system.conf, and in which the server log files and transaction logs are written.

While convenient for new users and small examples, this is not a recommended configuration for enterprise-level GemStone production environments. The *System Administration Guide* provides much more guidance on setting up a scalable, multi-user GemStone configuration.

The distribution installation is setup for this configuration; all that is needed is to copy an extent into the data directory.

1. Copy `$GEMSTONE/bin/extent0.dbf` to `$GEMSTONE/data/`.

   ```
   os$ cp $GEMSTONE/bin/extent.dbf $GEMSTONE/data/
   ```

2. chmod the extent file to have user and group write.

   ```
   os$ chmod ug+w $GEMSTONE/data/extent0.dbf
   ```

3. Start the stone.

   ```
   os$ startstone
   ```

   This will start the stone with the default name **gs64stone**.

4. Start the NetLDI. This example starts the NetLDI in guest mode with captive account, using the default named gs64ldi.

   ```
   os$ startnetldi -g -a gsAsminUser
   ```

Using a NetLDI by Port number

Without the *nameOrPortNumber* argument, the NetLDI will be started with the default name **gs64ldi** and a randomly selected port. If you have not defined gs64ldi (see "Configure Services for a Named NetLDI" on page 11), you may use a port number; this requires an additional argument.

```
os$ startnetldi -g -a gsAsminUser  netldiNameOrPort
```

For example,

```
os$ startnetldi -g -a gsAsminUser 54321
```

Verifying your new repository by logging in

Verify your Stone and NetLDI by logging in with topaz, the command line environment. This verification is done on the same host as the stone, with the gemstone environment setup. You will need to set login parameters, and execute the topaz **login** command.

You should ensure that your envionment is setup as described on page 10, that is, with the GEMSTONE environment and path set up.

For example:

```
os$ export GEMSTONE=InstallDir/GemStone64Bit3.7.2-platform
os$ export PATH=$GEMSTONE/bin:$PATH

os$> topaz
<startup headers>
topaz> set stone gs64stone user DataCurator password swordfish
topaz> login
```

Configure and set up users in new repository

Users log into GemStone using a GemStone user account (UserProfile), which is distinct from the UNIX user account. Logging into GemStone normally requires first authentication for your unix userId, and then login to GemStone with the GemStone userId and password.

Note that there are ways to configure a streamlined authentication, as described in the The chapter entitled "User Accounts and Security" in the *System Administration Guide*.

Change Passwords for Administrative Accounts

GemStone comes with built-in administrative accounts, referred to as System accounts or System userProfiles. These accounts are used to perform system administration and maintenance operations.

- The **DataCurator** account is used to perform system administration tasks.
- The **SystemUser** account is ordinarily used only for performing GemStone system upgrades.
- The **GcUser** account is used by the garbage collection task, which runs automatically as a separate login.

The initial password for these administrative accounts is `swordfish`.

Access to each of these accounts should be restricted; you should always change the passwords for these accounts, to provide basic security for your application.

The chapter entitled "User Accounts and Security" in the *System Administration Guide* tells you how to change the passwords.

Install the default TimeZone

The GemStone/S 64 Bit extents come with the time zone of 'America/Los_Angeles' installed as the default.

If you are in another time zone, login to GemStone as SystemUser and execute `TimeZone installOsTimeZone` to install the time zone configured in the OS, or `TimeZone installNamedZone:` to install another time zone by the zoneinfo (Olson) name.

You may also edit the file `installtimezone.txt` in the `$GEMSTONE/upgrade` directory, and file it in as SystemUser.

GemStone User Accounts

All users who log into GemStone should have a GemStone user account. While possible, it is not recommended to do development as DataCurator.

The chapter entitled "User Accounts and Security" in the *System Administration Guide* provides information on how create accounts for GemStone users, and the options for authentication. This task can be done by executing Smalltalk code using topaz, or tools included the GemBuilder for Smalltalk (GBS) product. See the *GemBuilder for Smalltalk Users's Guide* for information on the GUI tools in GBS.

Before login, Unix users should:

- Set the environment variable `GEMSTONE` to the *full pathname* (starting with a slash) of the GemStone/S 64 Bit 3.7.2 directory.
- Update their path to include the `$GEMSTONE/bin` directory.
- Optionally, update the man path (MANPATH variable) to include the `$GEMSTONE/doc` directory. GemStone provides man pages for utility functions.

For example:

```
os$ export GEMSTONE=InstallDir/GemStone64Bit3.7.2-platform
os$ export PATH=$GEMSTONE/bin:$PATH
os$ export MANPATH=$MANPATH:$GEMSTONE/doc
```

If the user will use GemStone frequently, consider adding these steps to the user's login shell initialization file.

Chapter

3 Upgrading from GemStone/S 64 Bit 3.3.x or later

This chapter describes how to upgrade an existing GemStone/S 64 Bit 3.3.x, 3.4.x, 3.5.x, or 3.6.x installation to GemStone/S 64 Bit version 3.7.2.

To upgrade from GemStone/S 64 Bit version 3.2.x and earlier, you must first upgrade to a more recent version, and then upgrade to v3.7.2.

If you are using GemBuilder for Smalltalk (GBS), you also need to upgrade the client libraries that are used by GBS. You may also need to upgrade your version of GBS; versions of GBS earlier than 8.8.1 and 5.4.7 may be usable, but are not fully supported with v3.7.2. See Chapter 5 for details.

Keyfiles

You may use a keyfile from 3.7 or any v3.7.x version with v3.7.2. Keyfiles from 3.6.x and earlier are not valid with v3.7.2. For more on keyfiles, see “Set the GemStone Keyfile” on page 10.

To acquire a keyfile for version 3.7.2, email keyfiles@gemtalksystems.com, or contact GemStone Technical Support, preferably providing your existing keyfile.

Keyfiles also manage access to GemConnect, GemBuilder for Java, and the X509-Secured GemStone feature. If you are using these add-on products, you must use a keyfile with the appropriate permissions.

Upgrade Procedure

The following list summarizes the steps necessary to perform the upgrade to GemStone/S 64 Bit v3.7.2.

NOTE

The following instructions use the version number **3.7.1** *to refer to the version you are upgrading from, and version number* **3.7.2** *indicate the target version you are upgrading to.*

Prior to Upgrade in existing application

1. Check for use of deprecated methods

Verify that your application does not invoke any methods that were deprecated in previous releases, by enabling error or logging on deprecation in your existing repository. Deprecated methods are subject to removal in major releases; finding them before upgrade allows the deprecation messages to provide replacement instructions.

For details on finding deprecated methods, refer to the *Programming Guide for GemStone/S 64 Bit*.

2. File out modifications to GemStone classes

File out any modifications or additions you made to GemStone/S 64 Bit kernel class methods. For more information about fileout, see the *GemStone/S 64 Bit Topaz Programming Environment*.

You will need to carefully compare these changes with GemStone/S 64 Bit 3.7.2 kernel methods, and refer to the *Release Notes* for version 3.7.2 to determine whether your changes are still necessary or appropriate.

CAUTION

Any changes that you have made to the GemStone/S 64 Bit kernel classes will be lost during upgrade; you MUST file these out in order to preserve the changes in version 3.7.2.

Prepare for Upgrade

1. Install and configure GemStone/S 64 Bit 3.7.2

Install GemStone/S 64 Bit 3.7.2 to a new installation directory, separate from the installation directory for version 3.7.1, as described in Chapter 1, starting on page 7.

Configure GemStone/S 64 Bit 3.7.2 the way you expect to use it — that is, with the appropriate extent locations and sizes.

If you copy the configuration files from your previous version to the version 3.7.2, be sure to review any changes in configuration parameters to determine if changes are needed.

You should ensure that adequate space is available for extents, transaction logs, and a backup during the upgrade. You must provide space for the extents and transaction logs for both repositories, the old and the new.

2. Reset SystemUser password

Log in to the version 3.7.1 system as a user with OtherPassword privilege, such as DataCurator, and reset the SystemUser password to 'swordfish':

```
topaz 1> printit
(AllUsers userWithId: #SystemUser) password: 'swordfish' .
System commitTransaction.
%
```

The upgrade script logs in with the SystemUser account and the default password, and resets the password for DataCurator and GcUser.

3. Stop user activity

Log in to the version 3.7.1 system as a user with SessionAccess and SystemControl privileges, such as DataCurator, and halt all user activity on the repository.

```
topaz 1> exec System stopUserSessions %
```

4. Shut down the repository

You may now shut down the Stone. At the UNIX command line:

stopstone *stone371*

where *stone371* is the name of the version 3.7.1 stone on this machine. The repository must be cleanly shut down to avoid needing recovery when it is restarted with the new version's executables.

5. Set up the version 3.7.2 environment.

Set the environment variables required for the upgrade. For example:

export GEMSTONE=*InstallDir372*
export PATH=$GEMSTONE/bin:$PATH
export upgradeLogDir=*tempDir*

where *InstallDir372* is the GemStone/S 64 Bit version 3.7.2 installation and *tempDir* is a temporary directory for which you have write permission.

6. Copy extent files

Copy your version 3.7.1 extent files into the location specified by the v3.7.2 configuration file option DBF_EXTENT_NAMES:

a. Identify the complete set of extent files that are used by your 3.7.1 stone. This can be found by examining the configuration file for the version 3.7.1 repository, looking for the last entry for DBF_EXTENT_NAMES.

b. The target location is the setting for `DBF_EXTENT_NAMES` in the version 3.7.2 installation. Copy each of these extent files to the target location.

For example:

```
cp dataDirFor371/extent0.dbf dataDirFor372
cp dataDirFor371/extent1.dbf dataDirFor372
cp dataDirFor371/extent2.dbf dataDirFor372/
```

Before upgrading, ensure that there are no transaction logs from a previous version of GemStone/S 64 Bit in any of the transaction log locations specified in the configuration file that will be used by version 3.7.2.

Perform the Upgrade

1. Start the Stone

Start the 3.7.2 Stone on the 3.7.1 extents you just copied:

```
startstone stoneName372
```

2. Upgrade image

Ensure you are in a directory to which you have write permission, and run the upgrade script.

The upgrade is performed by the script `upgradeImage`. This script has optional switches to specify the stone name and to set to size of the GEM_TEMPOBJ_CACHE_SIZE used for the upgrade process.

`upgradeImage [-h] [-c` *cacheSize*`] [-s` *stoneName*`]`

`-h` prints this usage information.

`-c` *cacheSize* sets the size of the GEM_TEMPOBJ_CACHE_SIZE; if this is not used, the script will default to use a value of 100000.

`-s` *stoneName* sets the name of the running stone to upgrade; if this option is not used, the script will default to **gs64stone.**

For example,

```
upgradeImage -s stoneName372
```

The script will prompt you to press the return key to begin.

The script invokes subordinate scripts to complete the upgrade. The upgrade process will take some time. You can examine the progress, if desired, by examining the file `$upgradeLogDir/upgradeImage*.out`.

The script should complete with the message:

```
Upgrade completed. No errors detected.
```

If not, please preserve the Stone log file and the contents of `$upgradeLogDir`. Contact your internal GemStone support person or GemStone Technical Support.

3. Restore System Account passwords

Log in to GemStone/S 64 Bit version 3.7.2 as DataCurator or SystemUser, and change the password for SystemUser, DataCurator, and GcUser to a secure password, such as the passwords used for these accounts in v3.7.1.

For example:

```
topaz 1> run
(AllUsers userWithId: 'SystemUser') password: '371Password'.
(AllUsers userWithId: 'GcUser') password: '371Password'.
(AllUsers userWithId: 'DataCurator') password: '371Password'.
System commitTransaction
%
```

where *371Password* is the account password used in version 3.7.1.

4. If you are upgrading from a 3.6.x version, recompile methods referencing instance of reimplemented class definitions

GsHostProcess and JsonParser had new implementations in v3.7 that affect some applications.

This step only applies if you are upgrading from 3.6.x or earlier.

GsHostProcess

If you have methods that encode a reference to the GsHostProcess class (and you have not recompiled these methods after a previous upgrade to 3.6.4 or later), these methods must be recompiled. After upgrade, the compiled method will refer to the class ObsoleteGsHostProcess, which does not support new GsHostProcess primitives.

Affected instances can be found using an expression such as:

```
(ClassOrganizer newExcludingGlobals) referencesToObject:
      (ObsoleteClasses at: #ObsoleteGsHostProcess)
```

This returns instances of GsNMethod. Once these have been examined, recompile can be done using an expression of the form:

```
aGsNMethod recompileFromSource
```

Note that this recompiles using the SymbolList of the current session, which may affect other class references in the method.

If you refer to the GsHostProcess class by name or symbol, the correct class will be found in the SymbolList, and no further action is needed.

JsonParser

The JsonParser class has been replaced in v3.7 with a non-PetitParser based implementation, which is smaller and faster. The basic `parse:` API is the same, but PetitParser-specific methods are not present in the new implementation.

If you are using JsonParser to simply parse JSON, you do not need to do anything. The methods reference the older class (now JsonPetitParser); if you want faster JSON parsing, you can recompile methods with class references. Affected instances can be found using an expression such as:

```
(ClassOrganizer newExcludingGlobals) referencesToObject:
   (Globals at: #JsonPetitParser)
```

If you are using JsonParser as part of a PetitParser customization, references to the class by name must be updated to refer to 'JsonPetitParser' instead of 'JsonParser'.

References from compiled methods to the class will continue to work, but it is recommended to edit the source code, so that a future recompile does not

inadvertently introduce a reference to the new implementation, or fail due to a reference to an inherited instance variable.

GsDevKit Upgrade

If you are using the open-source Development Kit for GemStone/S 64 Bit (GsDevKit, also referred to as Seaside or GLASS), you will need to perform another step to upgrade your GsDevKit image. This step upgrades the GsDevKit base code, and you will also need to reload your application code. This upgrade process is only tested and supported for upgrades from 3.5.3 and later.

For details, see Chapter 4, starting on page 31.

When you have completed the GsDevKit upgrade, continue with the upgrade process and perform the following steps.

Post-upgrade Application Code Modifications

1. Reinstall any other GemStone products that modify kernel classes.

If you use GemConnect or GemBuilder for Java (GBJ), you must reinstall the appropriate version of these products into your repository at this time. Note that version 3.7.2 requires GBJ v3.2 or above, so GBJ upgrade is required.

To install, use the procedure in the *Installation Guide* for that product.

2. File in Kernel class changes

If you have modified any kernel class methods of the previous version or if you have added methods to kernel classes, carefully compare your changes with the changes in version 3.7.2 to see whether your changes are still necessary or appropriate. Carefully review the *Release Notes* for each intervening version, as well as examining code in the image. If the kernel class changes are still applicable, file in the changes, verify that errorcount is 0, and commit.

Make Backup

1. Make backup

At this point, you should create a full backup of the upgraded repository.

Configure GCI clients and GBS

1. Recompile User Actions

It is recommended to recompile and relink User Action libraries.

2. Configure GBS

If you are using GBS clients, ensure you are running a supported version of GBS and client Smalltalk. Some older versions of GBS may be usable but are not fully supported with GemStone/S 64 Bit v3.7.2.

Chapter 5, 'Configuring GBS for GemStone/S 64 Bit' provides the supported versions of GBS for use on this platform. If your GBS clients run on a different platform than your GemStone server, refer to the *Installation Guide* for that platform.

Follow the instructions in that chapter for configuring the correct library for GBS.

Chapter

4 Upgrading GLASS/GsDevKit Applications

This chapter describes the additional upgrade step that applies when upgrading an application that is using variants of the open-source Development Kit for GemStone/S 64 Bit (GsDevKit, referred to also as also as GLASS or Seaside) to GemStone/S 64 Bit v3.7.2. The term GsDevKit is used to collectively refer to any of these environments.

The process described here can be used to upgrade repositories using GLASS, GLASS1, the older GsDevKit environment, and tODE. There are a number of possible configurations and there may be additional setup required for some environments. For more background on these environments, see https://github.com/GsDevKit/GsDevKit_upgrade/blob/master/README.md#upgrading-glassgsdevkit-applications-to-gemstone-350

If you are using the most recent version, from github.com/GsDevKit/GsDevKit_home, then you may use the upgrade scripts provided there to perform the entire upgrade, rather than using the instructions in this *Installation Guide*.

The complete process for upgrading includes the GemStone standard upgrade, followed by additional GsDevKit upgrade. Upgrading GemStone is described in earlier chapters of this *Installation Guide*; Chapter 3. You will need to follow the steps in that chapter, which will note the point at which the GsDevKit upgrade takes place.

Upgrade Procedure

The GsDevKit upgrade occurs partway through a standard GemStone upgrade.

To upgrade a GsDevKit/GLASS application:

- First, install the new version of GemStone, and upgrade your repository, according to the instructions in Chapter 3.
- After the **upgradeImage** step, you will upgrade the GsDevKit application as described in this chapter.
- After the GsDevKit upgrade completes, continue with the remaining steps of the GemStone upgrade in Chapter 3.

1. Ensure that GemStone 3.7.2 is installed and your repository upgraded

You must first follow install version 3.7.2 and follow the instructions in Chapter 3, before you can upgrade your GsDevKit application. These instructions will let you know at which point you perform the GsDevKit upgrade.

You should also have confirmed that your application code has been updated as required. We recommend that you install your application code in a test GemStone/S 64 Bit v3.7.2 repository, and verify that your code is working correctly, making changes as necessary. Do this prior to upgrading the data in your application, to ensure the upgrade process will go smoothly.

2. If necessary, customize the upgrade instructions

There are many ways a GsDevKit or GLASS application may be built, and a variety of packages that can be loaded. The **upgradeSeasideImage** script will upgrade a hypothetical standard installation, but there may be customizations required in specific cases.

The upgrade is performed by an upgrade script. By default, this is `$GEMSTONE/upgrade/createGsDevKit_upgrade.topaz`.

The default upgrade script is a file containing topaz commands for the upgrade.

Example 4.1 Default upgrade instructions in createGsDevKit_upgrade.topaz

```
run
  UserGlobals
   at: #GsDevKit_Image_Upgrade
       put: (GsuAbstractGsDevKitUpgrade
          upgradeUserName: SeasideUpgradeUser).
System commitTransaction
%
```

In this script, SeasideUpgradeUser is an internal global that by default resolves to DataCurator.

Customizing upgrade

To create customized upgrade instructions, make a copy of this file, edit the copy, and pass the path to your customized upgrade script file as an argument to the **upgradeSeasideImage** script.

The following example shows a customized file.

Example 4.2 Example customized upgrade

```
run
UserGlobals
   at: #GsDevKit_Image_Upgrade
   put: ((GsuAbstractGsDevKitUpgrade
            upgradeUserName: 'DataCurator'
            upgradeSymbolDictName: #UserGlobals)
      bootstrapApplicationLoadSpecs: {
      { 'Metacello' .
        'github://dalehenrich/metacello-work:master/repository' } .
      { 'GLASS1' .
        'github://glassdb/glass:master/repository' .
        #( 'default' 'Base' 'Announcements') } .
      { 'Seaside3' .
        'github://SeasideSt/Seaside:master/repository' .
        #( 'CI') }
      } ).
System commitTransaction
%
```

Further information about LoadSpecs is provided in the comment for `$GEMSTONE/upgrade/createGsDevKit_upgrade.topaz`.

You may also refer to the example scripts on github, at github.com/GsDevKit/GsDevKit_upgrade/tree/master/bin.

3. Perform the Upgrade

The GsDevKit upgrade is performed by the script **upgradeSeasideImage**, which is located in the `$GEMSTONE/seaside/bin` subdirectory .

Prior to executing **upgradeSeasideImage**, if you need to do a customized upgrade, you should have set up the upgrade script as described above. This is provided a argument to this script.

upgradeSeasideImage [-c *tempObjCacheSize* **] [-s** *stoneName* **] [-u** *gemstoneUser* **]**
[-p *password* **] [-P** *pathToUpgradeScript* **] [-W] [-E]**

-c *tempObjCacheSize*
set the size of the GEM_TEMPOBJ_CACHE_SIZE; if omitted, default is 100000.

-E enable isNil optimization (leave the isNil selector optimized, and not recompilable). Base GemStone **upgradeImage** enables isNil optimization by default, but there are potential problems with isNil optimization in some Seaside projects. **upgradeSeasideImage** disables isNil optimization, unless the -E option is specified.
See https://github.com/GsDevKit/GsDevKit_upgrade/issues/30 for more information.

-u *gemstoneUser*
specify the GemStone user name, if seaside was installed as a user other than DataCurator. If omitted, defaults to DataCurator.

-p *password*
specify the password for *gemstoneUser*. If omitted, defaults to swordfish.

-P *pathToUpgradeScript*
path to customized to GsDevKit_upgrade instance creation script. If omitted, `$GEMSTONE/upgrade/createGsDevKit_upgrade.topaz`.

-s *stoneName*
set the name of the running stone to upgrade; if omitted, default is **gs64stone.**

-W enable GEM_LISTEN_FOR_DEBUG, and set up to allow debugging via **debuggem**.

For example,

```
os$ $GEMSTONE/seaside/bin/upgradeSeasideImage -s stoneName372
```

The script will prompt you to press the return key to begin.

The script should complete with the message:

```
Seaside Upgrade completed. No errors detected.
```

If you encounter errors in **upgradeSeasideImage**, and you are experienced at using **topaz**, you may use the **-W** argument, and follow the instructions in the *Topaz User's Guide* for v3.6 on how to use **debuggem** to attach to the upgrade process and debug the error.

4. Load your Application Code

After upgrade has successfully completed, reload your application code.

5. Complete the Upgrade Process

To complete the upgrade, return to Chapter 3 and complete the remaining upgrade steps.

Chapter

5 Configuring GBS for GemStone/S 64 Bit

The GemStone/S 64 Bit v3.7.2 server requires a compatible version of GBS. Versions of GemBuilder for Smalltalk (GBS) that are compatible with GemStone/S 64 Bit v3.7.2 do not support Smalltalk client applications running on Macintosh.

GemStone/S 64 Bit v3.7.2 supports client Smalltalk/GBS applications running with VisualWorks on Linux and Windows, and with VAST (VA Smalltalk) running on Windows. For a table of all supported GBS and client Smalltalk platforms, see the *GemStone/S 64 Bit Release Notes* for v3.7.2.

For instructions for updating GBS clients that are running on Linux, see the *GemStone/S 64 Bit Installation Guide* for that platform; for GBS clients running on Windows, see the *GemStone/S 64 Bit Windows Client Installation Guide*.