G2 Java Socket Manager

User's Guide

Gensym Corporation

Version Gensym.2020.1, March 2020

Table of Contents

G2 Java Socket Manager, Version Gensym.2020.1

Ignite Technologies, Inc.
401 Congress Ave., Suite 2650
Austin, Texas USA 78701
Telephone: +1-800-248-0027
Email: success@ignitetech.com [mailto:success@ignitetech.com]

Preface

About this guide

This guide contains complete information about the G2/Java Socket Manager. This guide is designed to help users set up and use the socket manager.

Audience

To understand and use this guide, you must be familiar with the G2 real-time expert system. In addition, you should be familiar with basic Unix commands and the fundamentals of the HP OpenView and IBM NetView products, where appropriate. If you encounter G2 terms or concepts that you do not understand, refer to the *G2 Reference Manual*.

Conventions

This guide uses the following typographic conventions and conventions for defining system procedures.

Typographic

Convention Examples	Description
g2-window, g2-window-1, ws-top-level, sys-mod	User-defined and system-defined G2 class names, instance names, workspace names, and module names
history-keeping-spec, temperature	User-defined and system-defined G2 attribute names
true, 1.234, ok, “Burlington, MA”	G2 attribute values and values specified or viewed through dialogs
Main Menu > Start KB Workspace > New Object create subworkspace Start Procedure	G2 menu choices and button labels
conclude that the x of y . .	Text of G2 procedures, methods, functions, formulas, and expressions
new-argument	User-specified values in syntax descriptions
text-string	Return values of G2 procedures and methods in syntax descriptions

Convention Examples	Description
File Name, OK, Apply, Cancel, General, Edit Scroll Area	GUIDE and native dialog fields, button labels, tabs, and titles
File > Save Properties	GMS and native menu choices
workspace	Glossary terms
`c:\Program Files\Gensym\`	Windows pathnames
`/usr/gensym/g2/kbs`	UNIX pathnames
`spreadsh.kb`	File names
`g2 -kb top.kb`	Operating system commands
`public void main()` `gsi_start`	Java, C and all other external c

NOTE Syntax conventions are fully described in the G2 Reference Manual.

Procedure Signatures

A procedure signature is a complete syntactic summary of a procedure or method. A procedure signature shows values supplied by the user in italics, and the value (if any) returned by the procedure underlined. Each value is followed by its type:

g2-clone-and-transfer-objects
(*list*: class item-list, *to-workspace*: class kb-workspace,*delta-x*: integer, *delta-y*: integer)
→ transferred-items: g2-list

Related Documentation

Integrity

- Integrity Relelease Notes
- Integrity Demo Guide
- Integrity User's Guide
- Integrity Reference Manual
- Integrity with AutoDiscovery User's Guide
- SymCure User's Guide
- Integrity Utililies Guide
- DXI3DB-Primer
- G2-SNMP Bridges Installation and User's Guide

- Integrity G2/Java Socket Manager User's Guide
- Integrity SNMP User's Guide

G2 Core Technology

- G2 Bundle Release Notes
- Getting Started with G2 Tutorials
- G2 Reference Manual
- G2 Language Reference Card
- G2 Developer's Guide
- G2 System Procedures Reference Manual
- G2 System Procedures Reference Card
- G2 Class Reference Manual
- Telewindows User's Guide
- G2 Gateway Bridge Developer's Guide

G2 Utilities

- G2 ProTools User's Guide
- G2 Foundation Resources User's Guide
- G2 Menu System User's Guide
- G2 XL Spreadsheet User's Guide
- G2 Dynamic Displays User's Guide
- G2 Developer's Interface User's Guide
- G2 OnLine Documentation Developer's Guide
- G2 OnLine Documentation User's Guide
- G2 GUIDE User's Guide
- G2 GUIDE/UIL Procedures Reference Manual

G2 Developers' Utilities

- Business Process Management System User's Guide
- Business Rules Management System User's Guide
- G2 Reporting Engine User's Guide
- G2 Web User's Guide
- G2 Event and Data Processing User's Guide
- G2 Run-Time Library User's Guide

- G2 Event Manager User's Guide
- G2 Dialog Utility User's Guide
- G2 Data Source Manager User's Guide
- G2 Data Point Manager User's Guide
- G2 Engineering Unit Conversion User's Guide
- G2 Error Handling Foundation User's Guide
- G2 Relation Browser User's Guide

Bridges and External Systems

- G2 ActiveXLink User's Guide
- G2 CORBALink User's Guide
- G2 Database Bridge User's Guide
- G2-ODBC Bridge Release Notes
- G2-Oracle Bridge Release Notes
- G2-Sybase Bridge Release Notes
- G2 JMail Bridge User's Guide
- G2 Java Socket Manager User's Guide
- G2 JMSLink User's Guide
- G2 OPCLink User's Guide
- G2-PI Bridge User's Guide
- G2-SNMP Bridge User's Guide
- G2 WebLink User's Guide

G2 JavaLink

- G2 JavaLink User's Guide
- G2 DownloadInterfaces User's Guide
- G2 Bean Builder User's Guide

G2 Diagnostic Assistant

- GDA User's Guide
- GDA Reference Manual
- GDA API Reference

Customer Support Services

You can obtain help with this or any Gensym product from Gensym Customer Support. Help is available online, by telephone and by email.

To obtain customer support online:

→ Access Ignite Support Portal at https://support.ignitetech.com.

You will be asked to log in to an existing account or create a new account if necessary. Ignite Support Portal allows you to:

- Register your question with Customer Support by creating an Issue.
- Query, link to, and review existing issues.
- Share issues with other users in your group.
- Query for Bugs, Suggestions, and Resolutions.

To obtain customer support by telephone or email:

→ Use the following numbers and addresses:

United States Toll-Free +1-855-453-8174

United States Toll +1-512-861-2859

Email support@ignitetech.com [mailto:support@ignitetech.com]

Chapter 1. Sockets and the G2 Java Socket Manager

Provides an introduction to sockets and the G2-JAVA Socket Manager.

1.1. Introduction

An understanding of sockets is prerequisite to effectively understanding and using the G2-Java Socket Manager. This chapter discusses how sockets work and introduces the G2-Java Socket Manager.

1.2. Introduction to Sockets

Transmission of data over a network can be complicated. Transmission at the network level requires several actions:

1. The sender splits a segment of user data into packets.
2. Each packet is transmitted over the network independently of the other data packets.
3. The receiver of the data reassembles the packets of data in the correct order to recreate the original segment of data. To do so, it must take in account any missing data or any data which arrived out of order.

A socket is an abstraction that allows a user to treat a network connection like an input/output stream. Bytes can be written to and read from a socket, and to the user the operation appears like a file I/O. Sockets shield the user from low-level operations of the network such as media types, packet retransmission, packet sizes, etc.

Essentially, a socket is a connection between two hosts. Socket functions can include:

- Connecting to a remote machine.
- Sending data.
- Receiving data.
- Closing a connection.
- Binding to a port.

- Listening for incoming data.
- Accepting connections from remote machines on the bound port.

1.3. JAVA Language Sockets

The JAVA language distinguishes between two types of sockets. A server socket has all of the capabilities listed above, a client socket has only the first four (opening and closing a connection and sending and receiving data).

1.4. JAVA Client Sockets

A client socket normally has the following life cycle:

- Create the socket.
- Try to connect to a server socket on a remote host. When the connection is made, it is full-duplex (data can travel both ways simultaneously).
- Exchange data.
- Close the connection. The server or the client side can close the connection. Java Server Sockets

A server socket normally has the following life cycle:

- Create the server socket on a particular port.
- Listen for incoming connection requests on that port - a successfully solicited connection results in a socket connecting the client and server.
- Exchange data.
- Close the connection. The server or the client side can close the connection.
- The server socket returns to step 2 - listen for incoming connection requests.

NOTE	A server socket can handle multiple client connections simultaneously: it accepts connection requests while servicing already-connected clients.

1.5. G2 Java Socket Manager Process and the Socket Manager Knowledge Base

The socket manager knowledge base `gsockman.kb` coupled with the Java socket manager process (JavaG2SocketManager) provide all of the functionality required for G2 to interact with sockets.

The Java process:

- Allows the user to establish a link to a G2 process on a specified host and port number.
- Is responsible for creating the sockets at the request of the G2 process, maintaining these sockets until G2 issues close socket requests (at which point it takes care of closing and cleaning up references to the Java socket objects).
- Takes care of passing any data that conforms to the requirements for pattern matching into G2.

1.5.1. Starting the Java Process

The following command in a DOS command window starts the Java process:

```
SocketManager [-port <port number>] [-debug <1-5>]
```

The `port` default is 22044. The port number specified here is a port for communication between G2 and the Java process. It is NOT available as a socket once G2 is connected, nor does it have any connections to the sockets created by G2. Port numbers cannot be shared.

NOTE Although you can connect a client socket to this port number, this connection is not valid for passing data and therefore should not be used.

The `debug` level specified indicates to the process which informational messages to display to the process window. If the debug level is unspecified, it is set to 0. Available levels are:

Table 1. Debug-levels

Level	Symbol	Description
0	MANDATORY	mandatory messages (can't be turned off)
1	FATAL_ERROR	fatal bridge errors
2	NON_FATAL_ERROR	non fatal bridge errors
3	G2_JAVA_CONNECTION	errors between G2 and java - remote procedure call errors
4	SOCKET_INFORMATION	informational Java process methods messages
5	TROUBLESHOOT	troubleshooting messages

The debug option can be overridden in G2. The process displays messages at the debug level that is set and also displays all messages at lower levels. For example, choosing level 4 displays messages at levels 4, 3, 2, 1 and 0.

After the Java process starts, it waits for a connection from G2. After making a connection with G2, the process waits for socket creation instructions from G2. After any sockets are established, the process handles reading data from the sockets and passing it to G2, as

appropriate. (G2 sockets define message delimiters that request that data only within the delimiters pass through.) G2 can also send data over a socket through the Java process.

The JAVA-G2 socket manager kb has three object definitions object definitions to create a management scenario for a set of sockets:

- Socket Manager Interface
- G2 Server Socket
- G2 Client Socket

1.6. Socket Manager Interface

This object initiates and manages the connection between G2 and the Socket Manager Java process. If a Java socket manager process is running on the host/port specified, it will connect and be considered *live*. See Object g2-java-socket-manager-interface for details.

1.7. G2 Server Socket

This object represents a Java server socket. The user *must* connect the g2 server socket object instance to a *live* socket manager interface and select the user menu option `g2-java-socket-create-connection-channel`, or the call the Procedure g2-java-socket-manager-open-socket before G2 will attempt to create the Java server socket. See description of the Object g2-java-server-socket for details.

1.8. G2 Client Socket

This represents a Java client socket. The user *must* connect the g2 client socket object instance to a *live* socket manager interface and select the user menu option `g2-java-socket-create-connection-channel` or the Procedure g2-java-socket-manager-open-socket before G2 will attempt to create the Java client socket. See description of the Object g2-java-client-socket for details.

Chapter 2. Socket Manager Object Definitions

Provides information on G2-Java Socket Manager object definitions.

2.1. Introduction

This chapter provides user information on G2-Java Socket Manager object definitions.

2.2. Socket Manager Object Definitions

The G2-Java Socket Manager includes the following object definitions:

- Object g2-java-socket-manager-interface
- Object g2-java-socket
- Object g2-java-server-socket
- Object g2-java-client-socket
- Object g2-java-socket-connection-channel

2.3. Object g2-java-socket-manager-interface

This object manages a link between G2 and the Java socket manager process. Note that only *one* link can exist between a Java socket manager process and G2. The G2 socket manager kb can support multiple socket manager interfaces, each reporting to its own Java socket manager process.

G2 server sockets and client sockets must be connected to a *live* interface object before they themselves can be connected.

NOTE	When socket objects in G2 are connected to the interface object, the G2 connection representation (the 'line' that connects the sockets to the interface object) does not relate to the actual opening or closing of sockets, but is a requirement for opening a socket. You *cannot* open a G2 socket that is not connected to an interface object.

When a new client/server is connected to an interface object in G2, the only two valid ways of making a socket connect request are:

- Selecting the menu choice Create-Socket to open the socket.
- Calling the Procedure g2-java-socket-manager-open-socket to open the socket.

Similarly, deleting the connection between a *live* interface object and a *live* socket object does not close the socket. The only two valid ways of closing a socket are:

- Selecting the user menu choice Close-Socket to close the socket.
- Initiating the Procedure g2-java-socket-manager-close-socket to close the socket.

NOTE A socket can still be closed even if the connection is broken.

Table 2. g2-java-socket-manager-interface

Attribute	Description
`remote-process-initialization-string`	Allows the user to override the Java socket manager process debug level. If set to "-debug", it forces the Java process into a debug level of TROUBLESHOOT.
Allowable values:	Any item or value.
Default value:	""
`gsi-connection-configuration`	Specifies the host and port to which to connect. The Java process is expected to be running on this host/port combination.
Allowable values:	Any item or value.
Default value:	""

2.4. Object g2-java-socket

This object is the parent class for server and client G2 representations of Java server and client sockets. The attributes described below are shared by both definitions.

Table 3. g2-java-socket

Attribute	Description
`regex-start-delimiter`	Message start delimiter - socket input is not to be passed to G2 until this message start is received. Default is a period (.), signifying any non-newline character received is a valid message start
Allowable values:	Any item or value

Attribute	Description
Default value:	"."
`regex-start-delimiter-case-insensitive`	Set to `TRUE` to allow case insensitive parsing of the START delimiter.
Allowable values:	Any `truth-value`.
Default value:	`TRUE`
`regex-start-delimiter-is-multi-line`	Affects the `$` and `^` REGEX symbols of the START delimiter. If `TRUE`, the `^` is treated as any start of line and the `$` as any end of line. If false the `^` is treated as the start of input and `$` as the end of input. In addition the period (.) meta character will not match newlines if `TRUE`.
Allowable values:	Any `truth-value`.
Default value:	`TRUE`
`regex-start-delimiter-is-extended-mask`	Tells the REGEX parser to ignore non- backslashed or embedded whitespaces in the START delimiter and allows the `#` to be a meta character introducing comments in the mask. This attribute allows long masks to be more legible.
Allowable values:	Any `truth-value`.
Default value:	`FALSE`
`regex-end-delimiter`	Message end delimiter - socket input is passed to G2 until this message end is received. Default is null ("") in which case there is no end character specified, and once the `regex-start-delimiter` has been found then the data stream into G2 will be anything received over the socket.
Allowable values:	Any item or value.
Default value:	""
`regex-end-delimiter-case-insensitive`	Same as `regex-start-delimiter-case-insensitive` but for the END delimiter.
Allowable values:	Any `truth-value`.
Default value:	`TRUE`

Attribute	Description
`regex-end-delimiter-is-multi-line`	Same as `regex-start-delimiter-is-multi-line` but for the END delimiter.
Allowable values:	Any `truth-value`.
Default value:	`TRUE`
`regex-end-delimiter-is-extended-mask`	Same as `regex-start-delimiter-is-extended-mask` but for the END delimiter
Allowable values:	Any `truth-value`
Default value:	`FALSE`
`receive-message-callback`	Specific user action for socket when message is received - use Procedure g2-java-example-receive-method as a template for a user defined procedure.
Allowable values:	Any item or value.
Default value:	NONE
`new-connection-channel-callback`	Specific user action for socket when new connection is made - use Procedure g2-java-example-new-connection-method as a template for a user defined procedure.
Allowable values:	Any item or value.
Default value:	NONE
`connection-channel-down-callback`	Specific user action for socket when connection is broken - use Procedure g2-java-example-connection-down-method as a template for a user defined procedure.
Allowable values:	Any item or value.
Default value:	NONE
`rcv-buffer-length User-specified`	buffer length for the Java socket receive buffer. Whether this value is accepted is platform- dependent, and even if it is set, the Java socket implementation is free to disregard it. If set to the default (0), the Java process uses the platform-dependent system defaults.
Allowable values:	Any item or value.
Default value:	0

Attribute	Description
`snd-buffer-length`	User-specified send buffer length for Java socket.send buffer. Whether this value is accepted is platform-dependent, and even if it is set, the Java socket implementation is free to disregard it. If set to the default (0) the Java process uses the platform-dependent system defaults.
Allowable values:	Any item or value.
Default value:	0
`enable-keep-alive`	Keep-alive setting for the Java socket. In Java bridge implementation, socket.setKeepAlive(true) is called when value of option is true. socket.setKeepAlive(true) will enable TCP keep-alive mechanism in the underlying socket connection. The interval used to check alive status is platform-dependent, normally this is 2 hours.
Allowable values:	Any truth-value.
Default value:	`FALSE`
`status`	When the socket creation is successful, G2 sets this to `ACTIVE`.
Allowable values:	`ACTIVE`, `INACTIVE`.
Default value:	`INACTIVE`
`strip-delimiters`	If `TRUE`, the message start and end delimiters will be stripped from any text passed to G2.
Allowable values:	Any `truth-value`.
Default value:	`FALSE`
`log-raw-socket-data`	If `TRUE` log the raw input stream to a file (i.e., all data that the socket sees). The file is named `RAW-[socket-id].txt` and is opened in the runtime directory by default, if it already exists it will be appended to - there is no option to specify any other file presently as it is intended as a debug function.
Allowable values:	Any `truth-value`.
Default value:	`FALSE`

Attribute	Description
`log-filtered-socket-data`	If `TRUE` log the filtered input stream to a file (i.e., all data that the socket sends to G2). Currently the file is named `FILTERED-[socket-id].txt` and is opened in the runtime directory by default, if it already exists it will be appended to - there is no option to specify any other file presently as it is intended as a debug function.
Allowable values:	Any `truth-value`.
Default value:	`FALSE`
`gfr-uuid`	See `gfr-item-with-uuid`.
Allowable values:	inherited
Default value:	<generated by GFR>.

2.5. Object g2-java-server-socket

This object is the G2 representation for Java server sockets - inherits all attributes from Object g2-java-socket and also has its own attributes:

Table 4. g2-java-server-socket

Attribute	Description
`local-port-number`	Local port that the server socket is listening on for connections.
Allowable values:	Any item or value.
Default value:	5000
`accept-timeout-in-ms`	The time (specified in milliseconds) the server socket will wait for a first connection before timing out - if 0 there is no timeout.
Allowable values:	Any integer.
Default value:	0

2.6. Object g2-java-client-socket

This object is the G2 representation for Java client sockets - inherits all attributes from Object g2-java-socket and has its own attributes also:

Table 5. g2-java-client-socket

Attribute	Description
`target-system`	The system to which the client is trying to connect. In order to make a connection, a server socket must be on this system listening on the specified port. If this field is left blank, the client looks for the connection on the local host.
Allowable values:	Any item or value.
Default value:	""
`remote-port-number`	The port number on the remote system to which the client will try to connect. There must be a server socket on this port waiting to accept connections.
Allowable values:	Any item or value.
Default value:	5000
`local-port-number`	The port on the machine local to the socket manager process that the client will open if the connection is successful.
Allowable values:	Any item or value.
Default value:	0

2.7. Object g2-java-socket-connection-channel

This object is the G2 representation of a socket connection.

Table 6. g2-java-socket-connection-channel

Attribute	Description
`cnx-channel-local-port`	The local port for this connection channel.
Allowable values:	Any item or value.
Default value:	0
`local-host`	The local host of this connection channel.
Allowable values:	Any text.
Default value:	""
`cnx-channel-remote-port`	The remote port of this connection channel.
Allowable values:	Any item or value.
Default value:	0

Attribute	Description
`remote-host`	The remote host of this connection channel.
Allowable values:	Any text.
Default value:	""
`num-messages-received`	The number of messages received by the connection channel since it was first connected. (Resetting the I/O for the socket does not reset this count.)
Allowable values:	Any item or value.
Default value:	0
`num-messages-sent`	The number of messages sent over the connection channel since it was first connected (resetting the I/O for the socket does not reset this count).
Allowable values:	Any item or value.
Default value:	0
`gfr-uuid`	See `gfr-item-with-uuid`.
Allowable values:	inherited.
Default value:	""

Chapter 3. Socket Manager User Menu Choices and Rules

Provides information on the G2-Java Socket Manager User Menu Choices.

3.1. Introduction

This chapter provides user information on the G2-Java Socket Manager user menu choices and rules.

3.2. User Menu Choices

User Menu Choices Menu choices are provided for the following objects:

- Menu for g2-java-socket-manager-interface
- Menu for any g2-java-socket
- Menu for g2-java-server-socket

3.3. Menu for g2-java-socket-manager-interface

Menu choices for the Object g2-java-socket-manager-interface are:

List-Connected-Sockets

List the connected servers and clients sockets for a *live* socket manager interface, according to the Java process.

List-All-Active-I/O-Threads

List the active Java process threads for each connected socket for a *live* socket manager interface according to the Java process. This is primarily for informational purposes only.

Each client socket has these threads:

1. One for receiving data
2. one for parsing the received data
3. One for writing data to be sent over the socket
4. one for actually outputting the data to be sent over the socket

The threads work in pairs to improve buffering capabilities: receiving and parsing is one and writing and outputing the other pair.

Each connected server has 1+4n threads associated with it (one for connection listening, and 4 for each connected client).

Each bound server socket with no client connections has one thread (for connection listening).

3.4. Menu for any g2-java-socket

Menu choices for any g2-java-socket are:

- Create-Socket
- List-Active-I/O-Threads-For-Socket
- Send-Data-Over-Socket
- Reset-Socket-I/O
- Close-Socket

3.4.1. Create-Socket

This user menu choice is available only for Object g2-java-server-socket and Object g2-java-client-socket connected to a *live* Object g2-java-socket-manager-interface. When selected, G2 asks the Java Socket Manager process to create the appropriate type of socket on the port specified.

For server sockets, the request should be successful if the port specified is not already bound (if some other process is connected to it). A successful connection is indicated in G2 by the socket object's turning green and its `STATUS` becoming `ACTIVE`.

For client sockets, the request should be successful if there is a server socket waiting to accept connections on the remote port specified. Note that the accepting server socket does not need to be a G2-controlled server socket. A successful connection is indicated in G2 by the socket object's turning green.

If an Object g2-java-client-socket is successful and creates and connects to a socket, or if an Object g2-java-server-socket creates a socket in response to a successfully initiated connection request, an Object g2-java-socket-connection-channel is created on the subworkspace of the g2 socket object to represent the connection. (If the workspace does not exist it is created.)

The connection channel object contains information on the connected hosts and their connection ports. It also keeps a count of the number of data packets sent or received by this socket connection. See Object g2-java-socket-connection-channel for more details.

3.4.2. List-Active-I/O-Threads-For-Socket

This user menu choice lists the active Java process threads for a *live* socket. This is primarily for informational purposes only.

Each client socket has four threads:

- One for receiving data and one for parsing the received data.
- One for writing data to be sent over the socket and one for actually outputting the data to be sent over the socket.

The threads work in pairs to improve buffering capabilities.

Each connected server has 1+4n threads associated with it (one for connection listening, and 4 for each connected client).

Each bound server socket with no client connections has one thread (for connection listening).

3.4.3. Send-Data-Over-Socket

This user menu choice is for testing of a *live* socket connection. The data saved in the variable send-data is sent over the socket. This variable data can be changed through a type-in box or procedure referencing it (see the demo kb for an example of this). The connection channel object of the socket should indicate that a message has been sent by incrementing the number of messages sent. If the receiving socket is also a g2 socket object (for example, a socket object in another g2 process), then the number of messages received of its connection channel object should be incremented also.

3.4.4. Reset-Socket-I/O

This user menu choice allows the user to to reset the input stream of the socket at any point. This action discards any data waiting to be passed into G2 on the receipt of a message end delimiter, and seeks the next start-delimiter. Note that this action is automatically performed whenever the user changes the start delimiter of the socket.

3.4.5. Close-Socket

Select this user menu choice to close an open socket. In G2, this option resets the socket object to status inactive, changes the color of the G2 socket object to coral, closes and deletes any connection channel objects on its subworkspace, and signals the Java process to close the socket. This has the effect of throwing away the Java object associated with the socket, freeing it up to be garbage-collected. Any reconnection on this socket results in the creation of a new Java socket object (assuming the port is still available), with no memory of what data had previously passed over the socket. Note that if a server socket is closed, any G2

client sockets associated with it are closed also.

3.5. Menu for g2-java-server-socket

The menu choice for any g2-java-server-socket is:

Display-Clients-for-Server-Socket

Displays a list of clients connected to this server socket.

3.6. G2-Java Socket Manager Rules

A rule has been defined to set the automatically reconnect for the socket manager interface to 30 seconds (default scan interval). This rule attempts to keep the connection alive if the connection is dropped between G2 and the Java Socket Manager. This rule can be found on the `RULES` subworkspace, which is accessible from the `JAVA-SOCKET-MANAGER` top-level workspace.

Chapter 4. Socket Manager APIs

Provides information on G2-Java Socket Manager APIs.

4.1. Introduction

This chapter describes the G2 Java Socket Manager application programmers' interface.

4.2. APIs

APIs for the G2/Java Socket Manager include:

- Procedure g2-java-socket-manager-open-socket
- Procedure g2-java-socket-manager-close-socket
- Procedure g2-java-socket-check-connection
- Procedure g2-java-example-connection-down-method
- Procedure g2-java-example-new-connection-method
- Procedure g2-java-example-receive-method

4.3. Procedure g2-java-socket-manager-open-socket

g2-java-socket-manager-open-socket
(*socketintf*: g2-java-socket-manager-interface , *socket*: g2-java-socket)

Table 7. g2-java-socket-manager-open-socket

Argument	Description
socketintf	The socket manager interface to which the socket is connected.
socket	The socket to open.

Call this API to programmatically connect a server/client socket. The socket interface specified must be connected to a socket manager Java process (connection status of 2). If the socket is already connected, no action is taken.

4.4. Procedure g2-java-socket-manager-close-socket

g2-java-socket-manager-close-socket
(*socketintf*: g2-java-socket-manager-interface , *socket*: g2-java-socket)

Table 8. g2-java-socket-manager-close-socket

Argument	Description
socketintf	The socket manager interface to which the socket is connected.
socket	The socket to close.

Call this API to programmatically disconnect a server/client socket. The socket interface specified must connected to a socket manager Java process (connection status of 2) If the socket is already disconnected, no action is taken.

4.5. Procedure g2-java-socket-check-connection

g2-java-socket-check-connection
(socket: g2-java-client-socket, send-msg: text, receive-msg: text, time-out: integer)
→ status: symbol

Table 9. g2-java-socket-check-connection

Argument	Description
socket	The g2 client socket object to check connection status.
send-msg	The message to send to remote host.
receive-msg	The message with which remote host should reply.
time-out	The maximum number of seconds to wait for the reply message.

Return value	Description
status	A symbol (`connected` or `disconnected` or `in-progress`) to indicate the connection status of the g2 client socket.

Call this API to programmatically check connection status of a java client socket. It will send `send-msg` to remote host and wait for remote host reply `receive-msg`. If `receive-msg` received within `time-out` seconds, symbol `connected` will be returned. Otherwise, symbol `disconnected` will be returned. Continuously calling this procedure for same java client socket before previous call returns will return symbol `in-progress` directly.

4.6. Procedure g2-java-example-connection-down-method

g2-java-example-connection-down-method
(*socket*: g2-java-socket , *socketcnxchannel*: g2-java-socket-connection-channel)

Table 10. g2-java-example-connection-down-method

Argument	Description
socket	The g2 socket object.
socketcnxchannel	The connection channel associated with the g2 socket object.

Example procedure to illustrate how the user can customize actions based on a g2 socket object losing a connection channel. Specify this as the receive message callback of the socket.

4.7. Procedure g2-java-example-new-connection-method

g2-java-example-new-connection-method
(*socket*: g2-java-socket , *socketcnxchannel*: g2-java-socket-connection-channel)

Table 11. g2-java-example-new-connection-method

Argument	Description
socket	The g2 socket object.
socketcnxchannel	The connection channel associated with the g2 socket object.

Example procedure to illustrate how the user can customize actions based on a g2 socket object receiving a new connection channel. Specify this as the new connection channel callback of the socket.

4.8. Procedure g2-java-example-receive-method

g2-java-example-receive-method
(*socket*: g2-java-socket , *cnxchannel*: g2-java-socket-connection-channel, rx: text)

Table 12. g2-java-example-receive-method

Argument	Description
socket	The g2 socket object.

Argument	Description
cnxchannel	The connection channel associated with this g2 socket object.
rx	The data received.

Example procedure to illustrate how the user can customize actions based on the receipt of data over a g2 socket object. Specify this as the connection channel down callback of the socket.

NOTE To specify your own methods for G2 actions on receipt of data, creation of connections, and closing connections, you must use the same procedure signatures.

Chapter 5. Regex Support

Provides information on the Regular Expression (Regex) support by G2-Java Socket Manager.

5.1. Support for Regular Expressions (Regex)

A regular expression (regex) is a pattern or template to be matched against a string. Regex support exists in socket manager to allow the user to accept data over a socket if it conforms to a certain pattern, specifically if it starts or ends with a given regular expression. Thus a stream of messages each with its own id following a predefined format can be pulled from a noisy data stream.

For example, if each message had an id of the format `"[0-9]{5}[a-z]{5} <some text> END"` and the data stream was:

```
89473hagyd Hello G2 END ignore this noise 34256huayd Goodbye G2 END ashkdhsa
```

The text passed to G2, assuming the start/end delimiters are `"[0-9]{5}[a-z]{5}"` and`"END"` would be:

```
89473hagyd Hello G2 END 34256huayd Goodbye G2 END
```

Since the regex `"[0-9]{5}[a-z]{5}"` specifies:

> "match exactly 5 digits followed by exactly 5 lowercase letters of the Roman alphabet"

And the end match `"END"` matches exactly those letters.

The purpose of this section is not to go into depth explaining regex, rather it is to explain the regex support available in Socket Manager. The user can reference excellent explanations of PERL regex support in "Learning PERL" by Schwartz & Christiansen (O'Reilly).

NOTE	This product includes software developed by the Apache Software Foundation (http://www.apache.org/). The REGEX support is provided by PERL support classes developed by the Apache Software Foundation. These classes are open source.

5.2. Single Character

The simplest pattern to match is a character that matches itself exactly. For example, "z" matches all instances of "z".

The "." matches any character except the new-line character. Thus this is the default for a socket start delimiter in G2.

5.3. Character Classes

A pair of open and close square brackets represents a character class match. Thus [abcxyz] matches any string containing the first three or last three letters of the Roman alphabet.

The dash character between the ends of the range can specify ranges. Thus [a-z] matches any lowercase Roman alphabet character, and [0-9] any digit.

To escape any character use the \. Thus [0-9\-] matches any digit OR the minus '- ' sign.

Negated ranges are specified with the ^, so [^aeiou] is any character BUT a lowercase vowel. [^\^] will match any character BUT a ^.

Chapter 6. Socket Manager Troubleshooting

Provides troubleshooting information on the G2 Java Socket Manager.

6.1. Introduction

troubleshooting This chapter includes a brief description of the common problems a user might encounter when using Socket Manager for data stream handling. It is not a complete list and is intended as a guide rather than a substitute for customer support.

When calling customer support, please have the following information available:

- Description of error.
- Socket Manager software version.
- JDK version.
- Actual error message text, if any.
- Description of actions that were taken previous to the error.
- Enough detail to duplicate the problem.

6.2. Java Process Will Not Start

Ensure that the Java class path is correct, and that the correct command is specified.

Check the installation guide for the correct install steps for your platform. Contact Gensym Customer Support if the problem persists.

When correctly started the Java process indicates that it is listening for G2 connections on the specified port number (or default 22044).

6.3. G2 Socket Manager Interface Object Will Not Connect to the Java Process

Ensure that the tcp-ip host name and port number are correctly specified on the gsi-

connection-configuration attribute of the interface object.

Ensure that G2 is started and that the Java process is running on this specified host/port combination. The Java process should indicate that it is listening for connections on that port number.

On a successful connection, the Object g2-java-socket-manager-interface turns green and its `gsi-interface-status` is set to 2. Contact Gensym Customer Support if the problem persists.

6.4. Server Socket Will Not Bind Specified Port

The port number is already bound by another process. Free up this socket before trying to reconnect. On UNIX based systems there may be a system introduced delay in freeing up this socket.

If the socket cannot be freed it is impossible to open a server socket on this connection.

6.5. Client Socket Will Not Connect to Specified Port

There is no server socket on the specified port. Create one or change the port to one where there is a known server socket.

6.6. Socket Indicates It is Connected But No Data Passes

This should be a VERY rare occurrence. However it is possible that if a socket is connected to remote socket which is closed uncleanly (uncleanly means any socket close down that does not send "end of stream" identifiers to its connected sockets), the remaining socket will not know of its remote socket's termination, and will therefore remain open. This occurrence has the following characteristics in G2:

- The socket remains green with a status of `ACTIVE`
- The socket threads remain active and are viewable through the menu choices
- According to the interface, this socket is a connected socket and its threads appear in a list of all active threads
- If it is a server socket, the client still remains in its list of connected clients
- Object g2-java-socket-connection-channel associated with this connection is still on the socket's subworkspace
- Data passing over this connection has no effect.

Action to take: Close the socket if it is a client using the user menu choice or the API to do so. If it is a server then *if absolutely necessary* delete the Object g2-java-socket-connection-channel representing the dead connection on the subworkspace of the server. To do this the user *must* know which connection object applies to this connection. Deleting a connection object in error will lead to error when data is received/sent for this connection. Leaving a disconnected connection object around is not a big problem, and it will be cleaned up automatically at the next socket manager interface reset/socket close.

6.7. Debug Options

Only use debug level greater than 3 if absolutely necessary. In particular, debug level TROUBLESHOOT will output information regarding the parsing of the input/output stream of the sockets and may flood the command window if the input/output stream is very chatty and possibly degrade the performance of the Java process.

6.8. Inconsistent Socket Connections - JavaSockets Are Not Connected in G2

This should be a *very rare* occurrence. Avoid using the port numbers in question and reset the interface at the next available opportunity. There should be no need to reset the Java process at this time. Contact Gensym Customer Support if the problem persists.

6.9. Thread Count for Socket Is Wrong

If any of these counts are incorrect, the Java process and the G2 socket object are out of synchronization, and the socket should be reset (closed and reopened). If this is not possible avoid using this port number and reset G2 at the next opportunity. Notify Gensym Customer Support of this occurrence.

6.10. Data Expected Does Not Appear in G2

Ensure that the REGEX patterns are specified correctly. If it is a server socket a telnet session can be opened on that port. Sending data over the server socket should cause the data to appear in the telnet session window (use the user menu choice to send data, and change the text of send-data procedurally or otherwise to check different data sequences). Likewise, typing into the telnet session window should cause the data to be transmitted to G2 IF it falls within the start/end delimiter requirements of the socket.

If the socket in question is a client socket then open a telnet session to the port that the client is connected to and if no data appears here then the problem lies with the server port.

If no data appears revert to the REGEX patterns "." and "" for start/end. Any characters sent over the remote socket should pass into the local socket and appear in G2.

If the problem persists contact Gensym Customer Support.

Index