# Guest Agent Protocol Reference Manual

QEMU version 2.10.2

This is the QEMU Guest Agent Protocol reference manual.

# Table of Contents

# 1 API Reference

General note concerning the use of guest agent interfaces:

"unsupported" is a higher-level error than the errors that individual commands might document. The caller should always be prepared to receive QERR_UNSUPPORTED, even if the given command doesn't specify it, or doesn't document any failure mode at all.

`guest-sync-delimited` [Command]

Echo back a unique integer value, and prepend to response a leading sentinel byte (0xFF) the client can check scan for.

This is used by clients talking to the guest agent over the wire to ensure the stream is in sync and doesn't contain stale data from previous client. It must be issued upon initial connection, and after any client-side timeouts (including timeouts on receiving a response to this command).

After issuing this request, all guest agent responses should be ignored until the response containing the unique integer value the client passed in is returned. Receival of the 0xFF sentinel byte must be handled as an indication that the client's lexer/tokenizer/parser state should be flushed/reset in preparation for reliably receiving the subsequent response. As an optimization, clients may opt to ignore all data until a sentinel value is receiving to avoid unnecessary processing of stale data.

Similarly, clients should also precede this **request** with a 0xFF byte to make sure the guest agent flushes any partially read JSON data from a previous client connection.

**Arguments:**

`id: int` randomly generated 64-bit integer

**Returns:** The unique integer id passed in by the client

**Since:** 1.1

`guest-sync` [Command]

Echo back a unique integer value

This is used by clients talking to the guest agent over the wire to ensure the stream is in sync and doesn't contain stale data from previous client. All guest agent responses should be ignored until the provided unique integer value is returned, and it is up to the client to handle stale whole or partially-delivered JSON text in such a way that this response can be obtained.

In cases where a partial stale response was previously received by the client, this cannot always be done reliably. One particular scenario being if qemu-ga responses are fed character-by-character into a JSON parser. In these situations, using guest-sync-delimited may be optimal.

For clients that fetch responses line by line and convert them to JSON objects, guest-sync should be sufficient, but note that in cases where the channel is dirty some attempts at parsing the response may result in a parser error.

Such clients should also precede this command with a 0xFF byte to make sure the guest agent flushes any partially read JSON data from a previous session.

**Arguments:**

`id: int` randomly generated 64-bit integer

**Returns:** The unique integer id passed in by the client

**Since:** 0.15.0

`guest-ping` [Command]

Ping the guest agent, a non-error return implies success

**Since:** 0.15.0

`guest-get-time` [Command]

Get the information about guest's System Time relative to the Epoch of 1970-01-01 in UTC.

**Returns:** Time in nanoseconds.

**Since:** 1.5

`guest-set-time` [Command]

Set guest time.

When a guest is paused or migrated to a file then loaded from that file, the guest OS has no idea that there was a big gap in the time. Depending on how long the gap was, NTP might not be able to resynchronize the guest.

This command tries to set guest's System Time to the given value, then sets the Hardware Clock (RTC) to the current System Time. This will make it easier for a guest to resynchronize without waiting for NTP. If no `time` is specified, then the time to set is read from RTC. However, this may not be supported on all platforms (i.e. Windows). If that's the case users are advised to always pass a value.

**Arguments:**

`time: int` (optional)
time of nanoseconds, relative to the Epoch of 1970-01-01 in UTC.

**Returns:** Nothing on success.

**Since:** 1.5

`GuestAgentCommandInfo` [Object]

Information about guest agent commands.

**Members:**

`name: string`
name of the command

`enabled: boolean`
whether command is currently enabled by guest admin

`success-response: boolean`
whether command returns a response on success (since 1.7)

**Since:** 1.1.0

`GuestAgentInfo` [Object]

Information about guest agent.

**Members:**

`version: string`
guest agent version

`supported_commands: array of GuestAgentCommandInfo`
Information about guest agent commands

**Since:** 0.15.0

`guest-info` [Command]

Get some information about the guest agent.

**Returns:** `GuestAgentInfo`

**Since:** 0.15.0

`guest-shutdown` [Command]

Initiate guest-activated shutdown. Note: this is an asynchronous shutdown request, with no guarantee of successful shutdown.

**Arguments:**

`mode: string` (optional)
"halt", "powerdown" (default), or "reboot"

This command does NOT return a response on success. Success condition is indicated by the VM exiting with a zero exit status or, when running with –no-shutdown, by issuing the query-status QMP command to confirm the VM status is "shutdown".

**Since:** 0.15.0

`guest-file-open` [Command]

Open a file in the guest and retrieve a file handle for it

**Arguments:**

`path: string`
Full path to the file in the guest to open.

`mode: string` (optional)
open mode, as per fopen(), "r" is the default.

**Returns:** Guest file handle on success.

**Since:** 0.15.0

`guest-file-close` [Command]

Close an open file in the guest

**Arguments:**

`handle: int`
filehandle returned by guest-file-open

**Returns:** Nothing on success.

**Since:** 0.15.0

`GuestFileRead` [Object]

Result of guest agent file-read operation

**Members:**

`count: int`
number of bytes read (note: count is **before** base64-encoding is applied)

`buf-b64: string`
: base64-encoded bytes read

`eof: boolean`
: whether EOF was encountered during read operation.

**Since:** 0.15.0

`guest-file-read` [Command]

Read from an open file in the guest. Data will be base64-encoded

**Arguments:**

`handle: int`
: filehandle returned by guest-file-open

`count: int` (optional)
: maximum number of bytes to read (default is 4KB)

**Returns:** `GuestFileRead` on success.

**Since:** 0.15.0

`GuestFileWrite` [Object]

Result of guest agent file-write operation

**Members:**

`count: int`
: number of bytes written (note: count is actual bytes written, after base64-decoding of provided buffer)

`eof: boolean`
: whether EOF was encountered during write operation.

**Since:** 0.15.0

`guest-file-write` [Command]

Write to an open file in the guest.

**Arguments:**

`handle: int`
: filehandle returned by guest-file-open

`buf-b64: string`
: base64-encoded string representing data to be written

`count: int` (optional)
: bytes to write (actual bytes, after base64-decode), default is all content in buf-b64 buffer after base64 decoding

**Returns:** `GuestFileWrite` on success.

**Since:** 0.15.0

`GuestFileSeek` [Object]

Result of guest agent file-seek operation

**Members:**

`position: int`
current file position

`eof: boolean`
whether EOF was encountered during file seek

**Since:** 0.15.0

`QGASeek` [Enum]

Symbolic names for use in `guest-file-seek`

**Values:**

`set` Set to the specified offset (same effect as 'whence':0)

`cur` Add offset to the current location (same effect as 'whence':1)

`end` Add offset to the end of the file (same effect as 'whence':2)

**Since:** 2.6

`GuestFileWhence` [Alternate]

Controls the meaning of offset to `guest-file-seek`.

**Members:**

`value: int`
Integral value (0 for set, 1 for cur, 2 for end), available for historical reasons, and might differ from the host's or guest's SEEK_* values (since: 0.15)

`name: QGASeek`
Symbolic name, and preferred interface

**Since:** 2.6

`guest-file-seek` [Command]

Seek to a position in the file, as with fseek(), and return the current file position afterward. Also encapsulates ftell()'s functionality, with offset=0 and whence=1.

**Arguments:**

`handle: int`
filehandle returned by guest-file-open

`offset: int`
bytes to skip over in the file stream

`whence: GuestFileWhence`
Symbolic or numeric code for interpreting offset

**Returns:** `GuestFileSeek` on success.

**Since:** 0.15.0

`guest-file-flush` [Command]

Write file changes bufferred in userspace to disk/kernel buffers

**Arguments:**

`handle: int`
: filehandle returned by guest-file-open

**Returns:** Nothing on success.

**Since:** 0.15.0

`GuestFsfreezeStatus` [Enum]

An enumeration of filesystem freeze states

**Values:**

`thawed` filesystems thawed/unfrozen

`frozen` all non-network guest filesystems frozen

**Since:** 0.15.0

`guest-fsfreeze-status` [Command]

Get guest fsfreeze state. error state indicates

**Returns:** GuestFsfreezeStatus ("thawed", "frozen", etc., as defined below)

**Note:** This may fail to properly report the current state as a result of some other guest processes having issued an fs freeze/thaw.

**Since:** 0.15.0

`guest-fsfreeze-freeze` [Command]

Sync and freeze all freezable, local guest filesystems. If this command succeeded, you may call `guest-fsfreeze-thaw` later to unfreeze.

**Note:** On Windows, the command is implemented with the help of a Volume Shadow-copy Service DLL helper. The frozen state is limited for up to 10 seconds by VSS.

**Returns:** Number of file systems currently frozen. On error, all filesystems will be thawed.

**Since:** 0.15.0

`guest-fsfreeze-freeze-list` [Command]

Sync and freeze specified guest filesystems. See also `guest-fsfreeze-freeze`.

**Arguments:**

`mountpoints: array of string` (optional)
: an array of mountpoints of filesystems to be frozen. If omitted, every mounted filesystem is frozen. Invalid mount points are ignored.

**Returns:** Number of file systems currently frozen. On error, all filesystems will be thawed.

**Since:** 2.2

`guest-fsfreeze-thaw` [Command]

Unfreeze all frozen guest filesystems

**Returns:** Number of file systems thawed by this call

**Note:** if return value does not match the previous call to guest-fsfreeze-freeze, this likely means some freezable filesystems were unfrozen before this call, and that the filesystem state may have changed before issuing this command.

**Since:** 0.15.0

`GuestFilesystemTrimResult` [Object]

**Members:**

`path: string`
path that was trimmed

`error: string` (optional)
an error message when trim failed

`trimmed: int` (optional)
bytes trimmed for this path

`minimum: int` (optional)
reported effective minimum for this path

**Since:** 2.4

`GuestFilesystemTrimResponse` [Object]

**Members:**

`paths: array of GuestFilesystemTrimResult`
list of `GuestFilesystemTrimResult` per path that was trimmed

**Since:** 2.4

`guest-fstrim` [Command]

Discard (or "trim") blocks which are not in use by the filesystem.

**Arguments:**

`minimum: int` (optional)
Minimum contiguous free range to discard, in bytes. Free ranges smaller than this may be ignored (this is a hint and the guest may not respect it). By increasing this value, the fstrim operation will complete more quickly for filesystems with badly fragmented free space, although not all blocks will be discarded. The default value is zero, meaning "discard every free block".

**Returns:** A `GuestFilesystemTrimResponse` which contains the status of all trimmed paths. (since 2.4)

**Since:** 1.2

`guest-suspend-disk` [Command]

Suspend guest to disk.

This command tries to execute the scripts provided by the pm-utils package. If it's not available, the suspend operation will be performed by manually writing to a sysfs file.

For the best results it's strongly recommended to have the pm-utils package installed in the guest.

This command does NOT return a response on success. There is a high chance the command succeeded if the VM exits with a zero exit status or, when running with –no-shutdown, by issuing the query-status QMP command to to confirm the VM status is `"shutdown"`. However, the VM could also exit (or set its status to `"shutdown"`) due to other reasons.

The following errors may be returned: If suspend to disk is not supported, Unsupported

**Notes:** It's strongly recommended to issue the guest-sync command before sending commands when the guest resumes

**Since:** 1.1

`guest-suspend-ram` [Command]

Suspend guest to ram.

This command tries to execute the scripts provided by the pm-utils package. If it's not available, the suspend operation will be performed by manually writing to a sysfs file.

For the best results it's strongly recommended to have the pm-utils package installed in the guest.

IMPORTANT: guest-suspend-ram requires QEMU to support the 'system_wakeup' command. Thus, it's **required** to query QEMU for the presence of the 'system_wakeup' command before issuing guest-suspend-ram.

This command does NOT return a response on success. There are two options to check for success:

1. Wait for the SUSPEND QMP event from QEMU
2. Issue the query-status QMP command to confirm the VM status is `"suspended"`

The following errors may be returned: If suspend to ram is not supported, Unsupported

**Notes:** It's strongly recommended to issue the guest-sync command before sending commands when the guest resumes

**Since:** 1.1

`guest-suspend-hybrid` [Command]

Save guest state to disk and suspend to ram.

This command requires the pm-utils package to be installed in the guest.

IMPORTANT: guest-suspend-hybrid requires QEMU to support the 'system_wakeup' command. Thus, it's **required** to query QEMU for the presence of the 'system_wakeup' command before issuing guest-suspend-hybrid.

This command does NOT return a response on success. There are two options to check for success:

1. Wait for the SUSPEND QMP event from QEMU

2. Issue the query-status QMP command to confirm the VM status is "suspended"

The following errors may be returned: If hybrid suspend is not supported, Unsupported

**Notes:** It's strongly recommended to issue the guest-sync command before sending commands when the guest resumes

**Since:** 1.1

### `GuestIpAddressType` [Enum]

An enumeration of supported IP address types

**Values:**

`ipv4` IP version 4

`ipv6` IP version 6

**Since:** 1.1

### `GuestIpAddress` [Object]

**Members:**

`ip-address: string`
IP address

`ip-address-type: GuestIpAddressType`
Type of `ip-address` (e.g. ipv4, ipv6)

`prefix: int`
Network prefix length of `ip-address`

**Since:** 1.1

### `GuestNetworkInterface` [Object]

**Members:**

`name: string`
The name of interface for which info are being delivered

`hardware-address: string` (optional)
Hardware address of `name`

`ip-addresses: array of GuestIpAddress` (optional)
List of addresses assigned to `name`

**Since:** 1.1

### `guest-network-get-interfaces` [Command]

Get list of guest IP addresses, MAC addresses and netmasks.

**Returns:** List of GuestNetworkInfo on success.

**Since:** 1.1

### `GuestLogicalProcessor` [Object]

**Members:**

`logical-id: int`
Arbitrary guest-specific unique identifier of the VCPU.

`online: boolean`
Whether the VCPU is enabled.

`can-offline: boolean` (optional)
Whether offlining the VCPU is possible. This member is always filled in by the guest agent when the structure is returned, and always ignored on input (hence it can be omitted then).

**Since:** 1.5

`guest-get-vcpus` [Command]

Retrieve the list of the guest's logical processors.

This is a read-only operation.

**Returns:** The list of all VCPUs the guest knows about. Each VCPU is put on the list exactly once, but their order is unspecified.

**Since:** 1.5

`guest-set-vcpus` [Command]

Attempt to reconfigure (currently: enable/disable) logical processors inside the guest.

The input list is processed node by node in order. In each node `logical-id` is used to look up the guest VCPU, for which `online` specifies the requested state. The set of distinct `logical-id`'s is only required to be a subset of the guest-supported identifiers. There's no restriction on list length or on repeating the same `logical-id` (with possibly different `online` field). Preferably the input list should describe a modified subset of `guest-get-vcpus`' return value.

**Arguments:**

`vcpus: array of GuestLogicalProcessor`
Not documented

**Returns:** The length of the initial sublist that has been successfully processed. The guest agent maximizes this value. Possible cases:

- 0: if the `vcpus` list was empty on input. Guest state has not been changed. Otherwise,
- Error: processing the first node of `vcpus` failed for the reason returned. Guest state has not been changed. Otherwise,
- < length(`vcpus`): more than zero initial nodes have been processed, but not the entire `vcpus` list. Guest state has changed accordingly. To retrieve the error (assuming it persists), repeat the call with the successfully processed initial sublist removed. Otherwise,
- length(`vcpus`): call successful.

**Since:** 1.5

`GuestDiskBusType` [Enum]

An enumeration of bus type of disks

**Values:**

`ide` IDE disks

`fdc` floppy disks

`scsi` SCSI disks

`virtio` virtio disks

`xen` Xen disks

`usb` USB disks

`uml` UML disks

`sata` SATA disks

`sd` SD cards

`unknown` Unknown bus type

`ieee1394` Win IEEE 1394 bus type

`ssa` Win SSA bus type

`fibre` Win fiber channel bus type

`raid` Win RAID bus type

`iscsi` Win iScsi bus type

`sas` Win serial-attaches SCSI bus type

`mmc` Win multimedia card (MMC) bus type

`virtual` Win virtual bus type `file-backed` virtual: Win file-backed bus type

`file-backed-virtual`
Not documented

**Since:** 2.2; 'Unknown' and all entries below since 2.4

`GuestPCIAddress` [Object]

**Members:**

`domain: int`
domain id

`bus: int` bus id

`slot: int` slot id

`function: int`
function id

**Since:** 2.2

`GuestDiskAddress` [Object]

**Members:**

`pci-controller: GuestPCIAddress`
controller's PCI address

`bus-type: GuestDiskBusType`
bus type

`bus: int` bus id

`target: int`
target id

`unit: int` unit id

**Since:** 2.2

`GuestFilesystemInfo` [Object]

**Members:**

`name: string`
disk name

`mountpoint: string`
mount point path

`type: string`
file system type string

`disk: array of GuestDiskAddress`
an array of disk hardware information that the volume lies on, which may be empty if the disk type is not supported

**Since:** 2.2

`guest-get-fsinfo` [Command]

**Returns:** The list of filesystems information mounted in the guest. The returned mountpoints may be specified to `guest-fsfreeze-freeze-list`. Network filesystems (such as CIFS and NFS) are not listed.

**Since:** 2.2

`guest-set-user-password` [Command]

**Arguments:**

`username: string`
the user account whose password to change

`password: string`
the new password entry string, base64 encoded

`crypted: boolean`
true if password is already crypt()d, false if raw

If the `crypted` flag is true, it is the caller's responsibility to ensure the correct crypt() encryption scheme is used. This command does not attempt to interpret or report on the encryption scheme. Refer to the documentation of the guest operating system in question to determine what is supported.

Not all guest operating systems will support use of the `crypted` flag, as they may require the clear-text password

The `password` parameter must always be base64 encoded before transmission, even if already crypt()d, to ensure it is 8-bit safe when passed as JSON.

**Returns:** Nothing on success.

**Since:** 2.3

`GuestMemoryBlock` [Object]

**Members:**

`phys-index: int`
: Arbitrary guest-specific unique identifier of the MEMORY BLOCK.

`online: boolean`
: Whether the MEMORY BLOCK is enabled in guest.

`can-offline: boolean` (optional)
: Whether offlining the MEMORY BLOCK is possible. This member is always filled in by the guest agent when the structure is returned, and always ignored on input (hence it can be omitted then).

**Since:** 2.3

`guest-get-memory-blocks` [Command]

Retrieve the list of the guest's memory blocks.

This is a read-only operation.

**Returns:** The list of all memory blocks the guest knows about. Each memory block is put on the list exactly once, but their order is unspecified.

**Since:** 2.3

`GuestMemoryBlockResponseType` [Enum]

An enumeration of memory block operation result.

**Values:**

`success` the operation of online/offline memory block is successful.

`not-found`
: can't find the corresponding memoryXXX directory in sysfs.

`operation-not-supported`
: for some old kernels, it does not support online or offline memory block.

`operation-failed`
: the operation of online/offline memory block fails, because of some errors happen.

**Since:** 2.3

`GuestMemoryBlockResponse` [Object]

**Members:**

`phys-index: int`
: same with the 'phys-index' member of `GuestMemoryBlock`.

`response: GuestMemoryBlockResponseType`
: the result of memory block operation.

`error-code: int` (optional)
: the error number. When memory block operation fails, we assign the value of 'errno' to this member, it indicates what goes wrong. When the operation succeeds, it will be omitted.

**Since:** 2.3

`guest-set-memory-blocks` [Command]

Attempt to reconfigure (currently: enable/disable) state of memory blocks inside the guest.

The input list is processed node by node in order. In each node `phys-index` is used to look up the guest MEMORY BLOCK, for which `online` specifies the requested state. The set of distinct `phys-index`'s is only required to be a subset of the guest-supported identifiers. There's no restriction on list length or on repeating the same `phys-index` (with possibly different `online` field). Preferably the input list should describe a modified subset of `guest-get-memory-blocks`' return value.

**Arguments:**

`mem-blks: array of GuestMemoryBlock`
Not documented

**Returns:** The operation results, it is a list of `GuestMemoryBlockResponse`, which is corresponding to the input list.

Note: it will return NULL if the `mem-blks` list was empty on input, or there is an error, and in this case, guest state will not be changed.

**Since:** 2.3

`GuestMemoryBlockInfo` [Object]

**Members:**

`size: int` the size (in bytes) of the guest memory blocks, which are the minimal units of memory block online/offline operations (also called Logical Memory Hotplug).

**Since:** 2.3

`guest-get-memory-block-info` [Command]

Get information relating to guest memory blocks.

**Returns:** `GuestMemoryBlockInfo`

**Since:** 2.3

`GuestExecStatus` [Object]

**Members:**

`exited: boolean`
true if process has already terminated.

`exitcode: int` (optional)
process exit code if it was normally terminated.

`signal: int` (optional)
signal number (linux) or unhandled exception code (windows) if the process was abnormally terminated.

`out-data: string` (optional)
base64-encoded stdout of the process

`err-data`: `string` (optional)
base64-encoded stderr of the process Note: `out-data` and `err-data` are present only if 'capture-output' was specified for 'guest-exec'

`out-truncated`: `boolean` (optional)
true if stdout was not fully captured due to size limitation.

`err-truncated`: `boolean` (optional)
true if stderr was not fully captured due to size limitation.

**Since:** 2.5

`guest-exec-status` [Command]

Check status of process associated with PID retrieved via guest-exec. Reap the process and associated metadata if it has exited.

**Arguments:**

`pid`: `int` pid returned from guest-exec

**Returns:** GuestExecStatus on success.

**Since:** 2.5

`GuestExec` [Object]

**Members:**

`pid`: `int` pid of child process in guest OS

**Since:** 2.5

`guest-exec` [Command]

Execute a command in the guest

**Arguments:**

`path`: `string`
path or executable name to execute

`arg`: `array of string` (optional)
argument list to pass to executable

`env`: `array of string` (optional)
environment variables to pass to executable

`input-data`: `string` (optional)
data to be passed to process stdin (base64 encoded)

`capture-output`: `boolean` (optional)
bool flag to enable capture of stdout/stderr of running process. defaults to false.

**Returns:** PID on success.

**Since:** 2.5

`GuestHostName` [Object]

**Members:**

`host-name: string`
Fully qualified domain name of the guest OS

**Since:** 2.10

`guest-get-host-name` [Command]

Return a name for the machine.

The returned name is not necessarily a fully-qualified domain name, or even present in DNS or some other name service at all. It need not even be unique on your local network or site, but usually it is.

**Returns:** the host name of the machine on success

**Since:** 2.10

`GuestUser` [Object]

**Members:**

`user: string`
Username

`domain: string` (optional)
Logon domain (windows only)

`login-time: number`
Time of login of this user on the computer. If multiple instances of the user are logged in, the earliest login time is reported. The value is in fractional seconds since epoch time.

**Since:** 2.10

`guest-get-users` [Command]

Retrieves a list of currently active users on the VM.

**Returns:** A unique list of users.

**Since:** 2.10

`GuestTimezone` [Object]

**Members:**

`zone: string` (optional)
Timezone name. These values may differ depending on guest/OS and should only be used for informational purposes.

`offset: int`
Offset to UTC in seconds, negative numbers for time zones west of GMT, positive numbers for east

**Since:** 2.10

`guest-get-timezone` [Command]

Retrieves the timezone information from the guest.

**Returns:** A GuestTimezone dictionary.

**Since:** 2.10

`GuestOSInfo` [Object]

**Members:**

`kernel-release: string` (optional)

- POSIX: release field returned by uname(2)
- Windows: version number of the OS

`kernel-version: string` (optional)

- POSIX: version field returned by uname(2)
- Windows: build number of the OS

`machine: string` (optional)

- POSIX: machine field returned by uname(2)
- Windows: one of x86, x86_64, arm, ia64

`id: string` (optional)

- POSIX: as defined by os-release(5)
- Windows: contains string "mswindows"

`name: string` (optional)

- POSIX: as defined by os-release(5)
- Windows: contains string "Microsoft Windows"

`pretty-name: string` (optional)

- POSIX: as defined by os-release(5)
- Windows: product name, e.g. "Microsoft Windows 10 Enterprise"

`version: string` (optional)

- POSIX: as defined by os-release(5)
- Windows: long version string, e.g. "Microsoft Windows Server 2008"

`version-id: string` (optional)

- POSIX: as defined by os-release(5)
- Windows: short version identifier, e.g. "7" or "20012r2"

`variant: string` (optional)

- POSIX: as defined by os-release(5)
- Windows: contains string "server" or "client"

`variant-id: string` (optional)

- POSIX: as defined by os-release(5)
- Windows: contains string "server" or "client"

**Notes:** On POSIX systems the fields `id`, `name`, `pretty-name`, `version`, `version-id`, `variant` and `variant-id` follow the definition specified in os-release(5). Refer to the manual page for exact description of the fields. Their values are taken from the os-release file. If the file is not present in the system, or the values are not present in the file, the fields are not included.

On Windows the values are filled from information gathered from the system.

**Since:** 2.10

`guest-get-osinfo` [Command]

Retrieve guest operating system information

**Returns:** `GuestOSInfo`

**Since:** 2.10

# Commands and Events Index

# Data Types Index