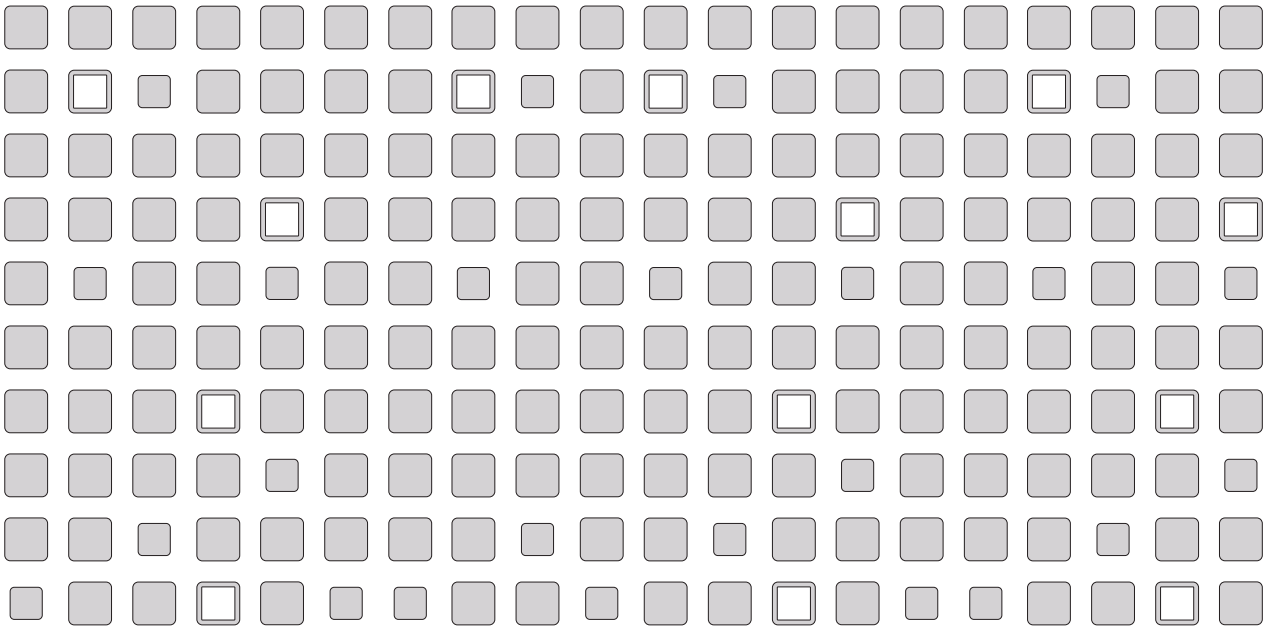


VERSION 1.0

# VMware Guest SDK

## Programming Guide



**VMware, Inc.**

3145 Porter Drive  
Palo Alto, CA 94304  
[www.vmware.com](http://www.vmware.com)

**Please note that you can always find the most up-to-date technical documentation on our Web site at <http://www.vmware.com/support/>.**

**The VMware Web site also provides the latest product updates.**

Copyright © 1998-2006 VMware, Inc. All rights reserved. Protected by one or more of U.S. Patent Nos. 6,397,242, 6,496,847, 6,704,925, 6,711,672, 6,725,289, 6,735,601, 6,785,886, 6,789,156 and 6,795,966; patents pending. VMware, the VMware "boxes" logo and design, Virtual SMP and VMotion are registered trademarks or trademarks of VMware, Inc. in the United States and/or other jurisdictions. Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation. Linux is a registered trademark of Linus Torvalds. All other marks and names mentioned herein may be trademarks of their respective companies.  
Revision: 20050718 Version: 1.0 Item: SDK-ENG-Q206-126

# 1

CHAPTER

## **VMware Guest SDK Programming Guide**

---

This manual provides the information a programmer needs to call the VMware Guest API from a program running in a VMware ESX Server virtual machine. It covers the following topics:

- [Overview of the VMware Guest API on page 4](#)
- [Programming to the VMware Guest API on page 6](#)

## Overview of the VMware Guest API

The VMware Guest API provides hooks that management agents and other software running in the guest operating system in a VMware ESX Server 3 virtual machine can use to collect certain data about the state and performance of the virtual machine. This is a read-only API. This means you can read data using this API, but you cannot send any control commands via this API.

The VMware Guest API is a complement to the VMware SDK.

The VMware Guest API provides fast access to resource management information with no need for authentication.

If you need to issue control commands, use the VMware SDK. For details, see the *VMware SDK Programming Guide* and the *VMware SDK Reference Guide*, available from the VMware Web site.

Using the VMware Guest API, you can monitor a number of different statistics about the virtual machine including:

- The amount of memory reserved for the virtual machine
- The amount of memory actually being used by the virtual machine
- The upper limit of memory available to the virtual machine
- The number of memory shares assigned to the virtual machine
- The maximum speed to which the virtual machine's CPU is limited
- The minimum reserved rate at which the virtual machine is allowed to execute; note that an idling virtual machine may consume CPU cycles at a much lower rate
- The number of CPU shares assigned to the virtual machine
- The elapsed time since the virtual machine was last powered on or reset
- CPU time scheduled on the ESX Server system for a particular virtual machine's CPU; combined with other available measurements, this allows you to estimate how fast the virtual machine's CPU is running compared to the host CPU
- Whether the API is able to provide accurate information (certain events, such as migrating a virtual machine with VMotion™, temporarily make it impossible to provide accurate information)

This information can be used by applications running in virtual machines to retrieve scheduling and resource utilization information about their environment. Using this information, the virtual machine can react to changes in the virtual environment immediately at the application layer.

## Installing Run-Time Components

The software required in the guest operating system is installed when you install VMware Tools. You must install the version of VMware Tools supplied with VMware ESX Server 3 or higher.

The run-time component is enabled by default. You can disable it by modifying an option in the virtual machine's configuration file. For details, see [Disabling the VMware Guest API on page 6](#).

## Supported Guest Operating Systems

The VMware Guest API is supported in any Windows or Linux guest operating system supported by ESX Server 3. To use the VMware Guest API, you must be running the virtual machine on an ESX Server 3 system.

# Programming to the VMware Guest API

The following sections provide the information you need to ensure that the VMware Guest API is enabled and to query the API for the information it makes available.

## Disabling the VMware Guest API

To disable the run-time component of the VMware Guest API, edit the configuration file for the virtual machine and add the following line or, if it already exists, update it to the following:

```
isolation.tools.guestlibGetInfo.disable = "TRUE"
```

The default value for this setting is "FALSE". The default setting enables the run-time component.

Reinstalling VMware Tools does not affect this setting. Therefore, if you disable the VMware Guest API, and then reinstall the tools, the VMware Guest API will continue to be unavailable until you change the `guestLibGetInfo.disable` configuration setting to "FALSE".

## Gathering Information from the VMware Guest API

The run-time component of the VMware Guest API is a single library file.

- In a Windows guest operating system, the library file is `vmGuestLib.dll`.
- In a Linux guest operating system, the library file is `libvmGuestLib.so`.

Use your program's standard methods to load the library in order to make the functions of the VMware Guest API available to your program.

## Understanding the VMware Guest API Data Types

The VMware Guest API uses several data types to facilitate access to virtual machine data..

Data Type	Description
<code>VMGuestLibHandle</code>	The reference to the data about the existing virtual machine.
<code>VMGuestLibSessionID</code>	The session ID is a unique identifier that changes after a virtual machine is migrated using VMotion, suspended and resumed, or reverted to a snapshot. Any of the events listed (migration with VMotion, suspend and resume, revert to snapshot) is likely to render invalid any information previously retrieved through this API. The session ID provides applications with a mechanism to detect those events and react accordingly — for example, by refreshing and resetting any state that relies on the validity of previously retrieved information.
<code>VMGuestLibError</code>	All VMware Guest API functions return an error message indicating the results of the call. For more information, see <a href="#">Understanding VMware Guest API Error Messages on page 10</a> .

## Making Calls to the VMware Guest API

The functions available in VMware Guest API are described in tables below.

### Open, Close, and Update Functions

The following functions are used to get new handles, release existing handles, and to update information:

Function	Description
VMGuestLib_OpenHandle	Gets a handle for use with other VMware Guest API functions.
VMGuestLib_CloseHandle	Releases a handle previously acquired with VMGuestLib_OpenHandle.
VMGuestLib_UpdateInfo	Updates information about the virtual machine stored at the VMGuestLibHandle.

### Accessor Functions

The functions in the following table retrieve information about a virtual machine. All the following functions return information about the attribute in question, but they also return a message about whether or not the function encountered an error. If the call completes successfully, the error returned is VMGUESTLIB\_ERROR\_SUCCESS, which means there was no error; if there is a problem with the call, information about the failure is returned. For more information on errors, see [Understanding VMware Guest API Error Messages on page 10](#).

Call VMGuestLib\_UpdateInfo once to refresh all statistics before calling an accessor functions or a series of accessors.

Function	Description
VMGuestLib_GetSessionId	Retrieves the ID for the current session after calling VMGuestLib_UpdateInfo(). The session ID is opaque and cannot be compared in any meaningful way with the session IDs from any other virtual machines. If VMGuestLib_UpdateInfo() has never been called, the return value is VMGUESTLIB_ERROR_NO_INFO.
VMGuestLib_GetCpuReservationMHz	Retrieves the minimum processing power in MHz reserved for the virtual machine. Assigning a cpuReservationMHz ensures that even as other virtual machines on a single host consume shared processing power, there is still a certain minimum amount reserved for this virtual machine.

Function	Description
VMGuestLib_GetCpuLimitMHz	<p>Retrieves the upper limit of processing power in MHz available to the virtual machine.</p> <p>Assigning a cpuLimitMHz ensures that this virtual machine never consumes more than a certain amount of the available processor power. By limiting the amount of processing power consumed, a portion of this shared resource is available to other virtual machines.</p>
VMGuestLib_GetCpuShares	Retrieves the number of CPU shares allocated to the virtual machine.
VMGuestLib_GetCpuUsedMs	<p>Retrieves the number of milliseconds during which the virtual machine has used the CPU. This value is the total amount of physical processor time used by the virtual machine including the time used by the guest operating system and the time used by virtualization code for tasks for this virtual machine. This value, in conjunction with elapsedMS, can be used to estimate effective virtual machine CPU speed. This value is a subset of elapsedMs.</p>
VMGuestLib_GetHostProcessorSpeed	Retrieves the speed of the ESX Server system's physical CPU in MHz.
VMGuestLib_GetMemReservationMB	<p>Retrieves the minimum amount of memory that is reserved for the virtual machine.</p> <p>Assigning a cpuReservationMB ensures that even as other virtual machines on a single host consume memory, there is still a certain minimum amount reserved for this virtual machine.</p>
VMGuestLib_GetMemLimitMB	<p>Retrieves the upper limit of memory that is available to the virtual machine.</p> <p>Assigning a cpuLimitMHz ensures that this virtual machine never consumes more than a certain amount of the available processor power. By limiting the amount of processing power consumed, a portion of this shared resource is available to other virtual machines.</p>
VMGuestLib_GetMemShares	Retrieves the number of memory shares allocated to the virtual machine.



Function	Description
VMGuestLib_GetMemMappedMB	Retrieves the amount of memory that is currently allocated to the virtual machine. Memory which is ballooned, swapped, or has never been accessed is excluded.
VMGuestLib_GetMemActiveMB	Retrieves the size of the memory the virtual machine is actively using — its estimated working set size.
VMGuestLib_GetMemOverheadMB	Retrieves the amount of "overhead" memory associated with this virtual machine that is currently consumed on the host system. Overhead memory is additional memory that is reserved for data structures required by the virtualization layer.
VMGuestLib_GetMemBalloonedMB	Retrieves the amount of memory that has been reclaimed from this virtual machine by the VMware memory balloon driver (also referred to as the "vmmemctl" driver.)
VMGuestLib_GetMemSwappedMB	Retrieves the amount of memory that has been reclaimed from this virtual machine by transparently swapping guest memory to disk.
VMGuestLib_GetMemSharedMB	Retrieves the amount of physical memory associated with this virtual machine that is copy-on-write (COW) shared on the host.
VMGuestLib_GetMemSharedSavedMB	Retrieves the estimated amount of physical memory on the host saved from copy-on-write (COW) shared guest physical memory.
VMGuestLib_GetMemUsedMB	Retrieves the estimated amount of physical host memory currently consumed for this virtual machine's physical memory. This is the same as (mapped memory) - (sharedSaved memory).
VMGuestLib_GetElapsedMs	Retrieves the number of milliseconds that have passed in the virtual machine since it last started running on the server where it is currently running. The count of elapsed time begins again any time the virtual machine is powered on, resumed, or migrated using VMotion.  This value counts milliseconds, regardless of whether the virtual machine is using processing power during that time. This value, in conjunction with cpuUsedMS, can be used to estimate effective virtual machine CPU speed. cpuUsedMS is a subset of this value.

Function	Description
VMGuestLib_GetResourcePoolPath	Retrieves the path name of the resource pool to which the virtual machine belongs on the ESX Server system where it is running.

For detailed information on ESX Server resource management, see the *VMware ESX Server Failover and Resource Management Guide*, available on the VMware Web site.

## Understanding VMware Guest API Error Messages

All VMware Guest API functions return an error message. In most cases, the error message returned will be VMGUESTLIB\_ERROR\_SUCCESS, meaning the API function completed successfully. In cases where the function is unable to complete its task, the error returned may provide information that is useful in diagnosing the problem. The following error codes can be returned by the VMware Guest API.

Error Code	Description
VMGUESTLIB_ERROR_SUCCESS	The function has completed successfully. This is the standard message that is returned after a function finishes.
VMGUESTLIB_ERROR_OTHER	An error has occurred. No additional information about the type of error is available.
VMGUESTLIB_ERROR_NOT_RUNNING_IN_VM	The program making this call is not running in a VMware virtual machine.
VMGUESTLIB_ERROR_NOT_ENABLED	The VMware Guest API is not enabled on this host, so these functions can not be used. For more information about how to enable the library, see <a href="#">Disabling the VMware Guest API on page 6</a> .
VMGUESTLIB_ERROR_NOT_AVAILABLE	The information you have requested is not available on this host.
VMGUESTLIB_ERROR_NO_INFO	VMGuestLib_UpdateInfo has not yet been called, so there is no information available to read from the data structure. Therefore, when an accessor function is called, there is no data to return.
VMGUESTLIB_ERROR_MEMORY	There is not enough memory available to complete the call.
VMGUESTLIB_ERROR_BUFFER_TOO_SMALL	The buffer is too small to accommodate the call. For example, when VMGuestLib_GetResourcePoolPath is called, if the path buffer is too small to accommodate the resource pool path, this error is returned. To resolve this error, allocate a larger buffer.
VMGUESTLIB_ERROR_INVALID_HANDLE	The handle you used is invalid. Ensure you have the correct handle and it has not been closed. You may need to create a new handle using VMGuestLib_OpenHandle().

Error Code	Description
VMGUESTLIB_ERROR_INVALID_ARG	One or more of the arguments passed to the function were invalid.

