What's New in the VI SDK 2.5?

The VMware Infrastructure (VI) SDK 2.5 supports many new VMware Infrastructure 3 features supported by ESX Server 3.5, VirtualCenter 2.5, and ESX Server 3i. The SDK 2.5 can also be used with ESX Server 3.0.x and VirtualCenter Server 2.0.x.

ESX Server 3.5, VirtualCenter Server 2.5, and ESX Server 3i support a new version of the VMware Infrastructure API—VI API 2.5—that provides new interfaces and data structures (managed objects, data objects, and so on) that comprise VMware infrastructure management components.

For example, VirtualCenter 2.5 provides a new patch-management feature, VMware Update Manager, which provides automated scanning (to determine appropriate patchsets) and patching of ESX Server systems (among other capabilities). The VI API 2.5 supports ESX Server patching with the addition of new managed object types and operations, specifically, the HostPatchManager managed object, with its ScanHostPatch_Task and InstallHostPatch_Task operations. (See Table A-1, "New Managed Objects Supported in VI API 2.5," on page 3 for a summary list.)

This chapter includes these topics:

- Support for New VI API Version (API 2.5)
- New Managed Object Types and Operations
- New Java Sample Applications

Support for New VI API Version (API 2.5)

In this release of the VI SDK, developers have access to two different API versions: VI API 2.0, and VI API 2.5. As in previous releases, the API is exposed as a secure Web service. The WSDL files defining each API version refer to different namespaces:

- The API 2.5 WSDL uses the namespace "vim25," which supports the API available on ESX Server 3.5, VirtualCenter Server 2.5, and ESX Server 3i Web services.
- The API 2.0 WSDL uses the namespace "vim2," which is the same as in previous releases of the SDK, including VI SDK 2.0.1. The VI API 2.0 is available on ESX Server 3.0.x, VirtualCenter Server 2.0.x, ESX Server 3.5, VirtualCenter Server 2.5, and ESX Server 3i.

How Does the VI API 2.5 Differ from Previous Releases?

The API 2.5 provides access to new data structures comprising the VMware Infrastructure object model—new managed objects, properties, and operations, available for the new features of the ESX Server and VirtualCenter Server.

In addition, in some cases, some of the existing managed objects, properties, and operations have been deprecated in favor of new capabilities. The differences can be summarized as follows:

- Completely new managed objects—for example, ExtensionManager, ViewManager, VirtualDiskManager, and HostPatchManager.

- New operations on existing managed objects, such as the Datacenter managed object type's new PowerOnMultiVM_Task, or the HostSystem managed object type's PowerDownHostToStandBy_Task and PowerUpHostFromStandBy_Task.
- New data objects, properties, and faults associated with both new and existing (VI API 2.0) data structures. For example, the new HostSystemHealthInfo data object type defines one of the properties of another new data object, HealthSystemRuntime. HealthSystemRuntime is used as a new property of HostRuntimeInfo, which is a data object defined in API 2.0.
- New properties on existing managed objects or data objects. For example, HealthSystemRuntime mentioned above, and the new alternateName and standardInquiry properties available on the ScsiLun data object.
- Deprecated managed objects, properties, or operations. For example, the QueryMemoryOverhead operation is deprecated in this release, in favor of the QueryMemoryOverheadEx.

Several new Java sample applications demonstrate using the new API, or handling differences between using the new API and its predecessor. See Table A-5 for a reference listing of new sample applications.

Also, in some cases, the object model has been refactored to provide a more robust foundation for future enhancement.

For example, a new managed object type—ExtensibleManagedObject—serves as the base class from which many other managed object types are derived. This change is transparent to client-application developers (meaning, you do not need to make changes to existing code). But you may notice the new structure as you navigate through the version 2.5 VI API Reference Guide.

See the VI API Reference Guide for more information. The API Reference Guide is available in the VI SDK 2.5 package, in this path:

```
\SDK\doc\ReferenceGuide
```

How Does the New API Affect Existing Applications?

Existing client applications that target ESX Server 3.0.x or VirtualCenter Server 2.0.x will continue to work, unchanged, against the ESX Server 3.5, VirtualCenter 2.5, and ESX Server 3i.

Can Client Applications Target Both ESX Server 3.0.x and ESX Server 3.5?

In a word, "yes." Client applications that use the VI API 2.0 (the vim2 WSDL) will be able to communicate with all target servers—ESX Server 3.5, VirtualCenter 2.5, ESX Server 3i, ESX Server 3.0.x, and VirtualCenter Server 2.0.x.

To leverage new features of ESX Server 3.5, VirtualCenter 2.5, and ESX Server 3i—the new HostPatchManager, for example—you must use the VI API 2.5 (the vim25 WSDL).

You can also create new client applications that support mixed server environments (ESX Server 3.5, ESX Server 3.0.x, VirtualCenter Server 2.5, VirtualCenter 2.0.x), and vary the behavior depending on the version. First, query the server to obtain the WSDL namespace version information. Then, use logic in your code to invoke operations or handle data types properly, appropriate for the server version. The VI SDK 2.5 includes several Java samples (in the \version sub-directory) that demonstrate how: The GetVersion sample is a utility class that obtains the version information from the server's WSDL. Other samples demonstrate how processing can vary, based on the server version. The version samples are discussed in more detail in "Handling Different API- and Object-Model-Versions" on page 8.

New Managed Object Types and Operations

The server-side object model provides several new managed objects that enable developers to leverage new features of ESX Server 3.5 and VirtualCenter 2.5.

Table A-1 summarizes the new managed objects available in the VI SDK 2.5. Of the Managed Object types listed in Table A-1, these four are used to model ServiceContent properties that return managed object references to singleton instances of the managed objects, as instantiated on the server:

- ExtensionManager
- FileManager
- ViewManager
- VirtualDiskManager

More information about these new services will be available soon, in an update to this New Features Summary.

Table A-1. New Managed Objects Supported in VI API 2.5

Managed Object	Description
ContainerView	View managed object type that facilitates monitoring the contents of a single container. Facilitates rendering object list within a specific container, for UI client applications. Extends (ManagedObjectView.)
ExtensibleManagedObject	Base interface for extensible managed object types.
ExtensionManager	Provides services to register and managed. Clients use the ExtensionManager, available in ServiceInstance, to access Extension objects.
FileManager	Provides operations for managing and manipulating files and folders on Datastore objects—for example, copying, deleting, and moving datastore files, and making directories on a Datastore. Experimental; subject to change.
HostBootDeviceSystem	Managed object type that describes the current system boot device configuration.
HostDateTimeSystem	Provides for NTP (network time protocol) and date and time configuration on a host. Information regarding the running status of the NTP daemon and functionality to start and stop the daemon is provided by the HostServiceSystem object.
HostFirmwareSystem	Provides access to the firmware of an ESX Server 3i host system, and enables ESX Server 3i configuration backup, restore, and reset. Experimental; subject to change.
HostHealthStatusSystem	Manages health state of the host, enabling checking state and resetting state.
HostPatchManager	Interface for scanning and patching an ESX Server using software updates available over the Web, from VMware.
InventoryView	View managed object type that enables browsing the inventory and tracking changes to open folders. Facilitates rendering tree-based inventory navigation for UI client applications. (Extends ManagedObjectView.)
ListView	View managed object type that enables modifying view content without adding or destroying filters. (Extends ManagedObjectView.)
ManagedObjectView	Base class for view objects that expose a set of ManagedObjects. Extended by ContainerView, InventoryView, and ListView.
View	Base class for session-specific view objects.
ViewManager	Provides access to managed objects that extend ViewManagedObject (base class), such as ContainerView, InventoryView, and ListView. These server-side view objects enable client applications to display subsets of properties for update.
VirtualDiskManager	Service for managing and manipulating virtual disks on datastores using URL or datastore path for source and target names.

For details about each of these new managed object types, see the VI API Reference Guide. The API Reference Guide is available in the VI SDK 2.5 package, in this path:

```
\SDK\doc\ReferenceGuide
```

New Operations

Table A-2 lists new operations available in the VI API 2.5.

Table A-2. New operations available in the VI API 2.5

Managed Object Type	Operation
ClusterComputeResource	RefreshRecommendation
ComputeResource	ReconfigureComputeResource_Task
Datacenter	PowerOnMultiVM_Task
EventManager	PostEvent
ExtensibleManagedObject	setCustomValue
ExtensionManager	SetPublicKey
	UnregisterExtension
	UpdateExtension
	FindExtension
	RegisterExtension
	GetPublicKey
FileManager	MoveDatastoreFile_Task
	MakeDirectory
	CopyDatastoreFile_Task
	DeleteDatastoreFile_Task
Folder	CreateClusterEx
HostBootDeviceSystem	UpdateBootDevice
	QueryBootDevices
HostDatastoreSystem	UpdateLocalSwapDatastore
HostDateTimeSystem	RefreshDateTimeSystem
	UpdateDateTime
	QueryAvailableTimeZones
	UpdateDateTimeConfig
	QueryDateTime
HostFirmwareSystem	RestoreFirmwareConfiguration
	ResetFirmwareToFactoryDefaults
	BackupFirmwareConfiguration
	QueryFirmwareConfigUploadURL
HostHealthStatusSystem	ResetSystemHealthInfo
	RefreshHealthStatusSystem
HostMemorySystem	ReconfigureVirtualMachineReservation
HostPatchManager	InstallHostPatch_Task
	ScanHostPatch_Task
HostSnmpSystem	SendTestNotification
	ReconfigureSnmpAgent

Table A-2. New operations available in the VI API 2.5 (Continued)

Managed Object Type	Operation
HostSystem	PowerDownHostToStandBy_Task
	AcquireCimServicesTicket
	PowerUpHostFromStandBy_Task
	QueryMemoryOverheadEx
	UpdateFlags
InventoryView	OpenInventoryViewFolder
	CloseInventoryViewFolder
LicenseManager	QuerySupportedFeatures
ListView	ResetListViewFromView
	ResetListView
	ModifyListView
PerformanceManager	QueryPerfCounterByLevel
ServiceInstance	RetrieveProductComponents
SessionManager	LoginBySSPI
	ImpersonateUser
	SessionIsActive
	LoginExtension
Task	SetTaskState
	UpdateProgress
TaskManager	CreateTask
View	DestroyView
ViewManager	CreateListView
	CreateInventoryView
	CreateContainerView
	CreateListViewFromView
VirtualDiskManager	ShrinkVirtualDisk_Task
	ZeroFillVirtualDisk_Task
	SetVirtualDiskUuid
	QueryVirtualDiskUuid
	ExtendVirtualDisk_Task
	CreateVirtualDisk_Task
	CopyVirtualDisk_Task
	DeleteVirtualDisk_Task
	MoveVirtualDisk_Task
	QueryVirtualDiskFragmentation
	DefragmentVirtualDisk_Task
	QueryVirtualDiskGeometry
	InflateVirtualDisk_Task
VirtualMachine	DefragmentAllDisks

See the VI API Reference Guide for more information. The API Reference Guide is available in the VI SDK 2.5 package, in this path:

```
\SDK\doc\ReferenceGuide
```

NOTE Some of the new managed object types or operations may be experimental, and subject to change, as noted in the VI API Reference Guide.

New Data Objects and Properties

In addition to new managed objects, the VI API 2.5 also includes many new data objects, properties, and enumerations associated with new (2.5) or existing (2.0) object model. The "Since VI API 2.5" label in the API Reference Guide identifies new managed object types, data object types, properties, and operations in this release.

Deprecated Properties and Types

Several properties and types available through the VI API 2.0 have been deprecated in VI API 2.5. Table A-3 lists the deprecated enumeration types, fault types, managed- and data-object properties, and managed object methods, and identifies the new feature that takes its place. For example, the QueryMemoryOverhead operation, available through HostSystem managed objects, is deprecated: QueryMemoryOverheadEx should be used for new development. (A Java sample, demonstrating using the new operation, is included in the SDK package. See "Handling Deprecated Operations" on page 9 for more information).

Table A-3. Deprecated Types, Properties, and Methods

Type	Name	Use instead...
Enumerated Types	LicenseManagerLicenseKey	Use QueryLicenseSourceAvailability to obtain an array of LicenseAvailabilityInfo data objects (which in turn, comprises LicenseFeatureInfo data objects and status information about number of licenses and availability).
	DrsRecommendationReasonCode	Use the RecommendationReasonCode enumerated type.
Fault Types	CpuCompatibilityUnknown	Use CpuIncompatible and its subclasses.
	IDEDiskNotSupported	Use the DeviceControllerNotSupported fault type.
Data Object Types	ClusterConfigInfo	Use the ClusterConfigInfoEx data object type.
	ClusterConfigSpec	Use the ClusterConfigSpecEx data object type.
	ClusterDrsRecommendation	Use ClusterRecommendation.
Managed Object Type Properties	ClusterComputeResource.configuration	Use ComputeResource.configurationEx
	ClusterComputeResource.drsRecommendation	Use ClusterComputResource.recommendation
	LicenseManager.featureInfo	Use QuerySupportedFeatures.

Table A-3. Deprecated Types, Properties, and Methods (Continued)

Type	Name	Use instead...
Managed Object Type Methods	ReconfigureCluster_Task	Use ReconfigureComputeResource_Task.
	RecommendHostsForVm	Use PowerOnMultiVM_Task. RecommendHostsForVm cannot make any recommendations if DRS cannot find the specified host in the cluster. With PowerOnMultiVM_Task, DRS atttempts to migrate virtual machines and power on hosts in standby mode, given the same conditions.
	DestroyDatastore	Do not use this method: DestroyDatastore throws ResourceInUse. Datastores are automatically removed when no longer in use, so this method is unnecessary.
	CreateCluster	Use CreateClusterEx.
	DeleteFile	Use DeleteDatastoreFile_Task (a method of the new FileManager managed object.
	QueryMemoryOverhead	Use QueryMemoryOverheadEx.
	DestroyNetwork	Do not use this method: DestroyNetwork throws ResourceInUse. Networks are automatically removed when no longer in use, so this method is unnecessary.
	CreatePerfInterval	Use UpdatePerfInterval. Historical intervals cannot be created.
	RemovePerfInterval	Use UpdatePerfInterval. Historical intervals cannot be removed.
Data Object Type Properties	ClusterDasVmConfigInfo.restartPriority	Use restartPriority. If this value is specified using both places, the value in restartPriority has precedence.
	ClusterDasVmConfigInfo.powerOffOnIsolation	Use isolationResponse. If a value is specified in both powerOffOnIsolation and isolationResponse, the value in isolationResponse takes precedence and the value in powerOffOnIsolation is ignored
	CustomizationWinOptions.deleteAccounts	Deleting user accounts as part of a customization routine is not supported as of VI API 2.5: the deleteAccounts property is ignored. To change the administrator password, set the administrator password to blank in the master vm. Sysprep will then be able to change the password to the one specified by the password.
	VirtualMachineFlagInfo.runWithDebugInfo	Use the new monitorType property (an enumeration comprising string values "debug," "release," and "stats").

New Java Sample Applications

In this release, you'll find new Java sample applications demonstrating new features of ESX Server 3.5 and VirtualCenter 2.5.

In addition, the Java samples have been improved over previous versions: Some of the code has been refactored; some of the Java packages have been reorganized. Other changes include:

- The supporting utility classes have been moved to their own package, apputil. Inside this package you'll find packages vim, vim25, and version. The version utilities facilitate determining the version of the WSDL on the Web service, and returning the version as a Java String. The version utility classes areused by DisplayNewProperties, GetVirtualDiskFiles, HostPowerOps, InstallHostPatch, and QueryMemoryOverhead.

See "Java Samples Summary" on page 10 for a reference table listing all the Java samples and package names. The next section discusses highlights of three key new samples.

Versioning, Durable LUN Names, and HTTP File Access Samples

Three new Java sample sets demonstrating several of the new features of ESX Server 3.5 and VirtualCenter 2.5 are discussed in this section:

- Handling Different API- and Object-Model-Versions
- Obtaining All Information about SCSI Devices
- Using HTTP for Cold Migration of Virtual Machine Files

Handling Different API- and Object-Model-Versions

As discussed in "How Does the VI API 2.5 Differ from Previous Releases?" on page 1, the VI API 2.5 supports many new managed object types, data object types, properties, and other data structures in the VMware Infrastructure object model. In addition, some object types that exist in VI API 2.0 have been deprecated in VI API 2.5.

NOTE The "deprecated" label on a feature (type, operation, property) means that a new, better way to meet a specific requirement exists (better than the deprecated feature). For example, QueryMemoryOverhead operation (of HostSystem) is deprecated as of VI API 2.5, and a new operation, QueryMemoryOverheadEx takes its place. Although client applications using deprecated features will continue to work, VMware recommends that developers use the new type, operation, or property (rather than the deprecation) for client applications using the VI API 2.5.

You can migrate existing client applications to work with ESX Server 3.5 or VirtualCenter 2.5, and you can target previous versions (ESX Server 3.0.x, VirtualCenter Server 2.0.x) as well—several Java samples demonstrating how to code for different targets are provided in the VI SDK package.

For starters, in a mixed target environment, obtaining version information about the server's API support is an important initial step. Since the API is exposed as a Web service, the Web service's WSDL file contains the information, and can be accessed at the Web service endpoint.

Obtaining Version Information from the WSDL File (GetVersion Sample)

The GetVersion sample (GetVersion.java) demonstrates how to obtain version information from the WSDL available at the HTTP or HTTPS server targets, both ESX Server and VirtualCenter Server systems. The version number is returned as a Java String (named `version`) that can be used by other Java classes, as a basis for subsequent processing. (You'll see that the other Java version samples use methods from GetVersion to obtain a value for the version String.)

Table A-4 lists the new samples included in this release that demonstrate using new features and handling deprecated object types, properties, or operations.

Table A-4. Version samples and what they demonstrate

Sample	Demonstrates...
DisplayNewProperties	Displaying various properties, depending on the WSDL version returned from the server. Behavior varies, depending the version returned. When connected to an ESX Server 3.5 or VirtualCenter Server 2.5 systems, DisplayNewProperties obtains (and displays) several new properties available through the API 2.5, including power state (runtime.powerState), boot duration (runtime.bootDuration), system health information (runtime.systemHealthInfo), and time information (config.dateTimeInfo).
GetVirtualDiskFiles	Using new properties of existing managed objects or data objects, depending on the WSDL version returned from the server. GetVirtualDiskFiles uses the HostDatastoreBrowser to find virtual disks on a specified virtual machine, to enable adding virtual disks to the VM. When connected to an ESX Server 3.5 system, GetVirtualDiskFiles invokes the SearchDatastoreSubFolders_Task using a data object (VmDiskFilleQueryFilter) that has a new property (controllerType). The result is that only those disks whose controller type matches the virtual machine controller will be shown.
HostPowerOps	Using new operations on existing managed objects. In the 2.5 object model, the HostSystem managed object type has two new operations, PowerDownHostToStandBy_Task, PowerUpHostFromStandBy_Task.
InstallHostPatch	Demonstrates using the HostPatchManager, available on ESX Server 3.5.
QueryMemoryOverhead	Using a new operation (QueryMemoryOverheadEx) in place of a deprecated operation.

Processing Logic Varies, Based on Version

The samples DisplayNewProperties, GetVirtualDiskFiles, PowerDownHostToStandBy, and QueryMemoryOverhead display different behavior depending on target server (and thus, the version of the API).

For example, when connected to an ESX Server 3.5, DisplayNewProperties lists power state (runtime.powerState), boot durations (runtime.bootDuration), system health information (runtime.systemHealthInfo), and time zone information (config.dateTimeInfo). Since none of these properties are supported through the VI API 2.0, when DisplayNewProperties runs against an ESX Server 3.0.x, these properties are not displayed.

As another example, when connected to an ESX Server 3.5 system, GetVirtualDiskFiles invokes the SearchDatastoreSubFolders_Task using a data object (VmDiskFilleQueryFilter) that has a new property (controllerType). The result is that only those disks whose controller type matches the virtual machine controller will be shown.

The PowerDownHostToStandBy sample (PowerDownHostToStandBy.java) demonstrates using a new operation (PowerDownHostToStandby).

Handling Deprecated Operations

Some operations available through the VI API 2.0 have been deprecated in VI API 2.5. For example, the QueryMemoryOverhead operation, available through HostSystem managed objects, has been deprecated. A new operation, QueryMemoryOverheadEx is recommended for new development. The QueryMemoryOverhead sample application (Java sample) demonstrates using the new QueryMemoryOverheadEx operation. QueryMemoryOverheadEx takes a VirtualMachineConfigInfo data object as one of its parameters: the VirtualMachineConfigInfo data object provides far more granularity in specifying the host details.

See the VI API Reference Guide for more information. The API Reference Guide is available in the VI SDK 2.5 package, in this path:

```
\SDK\doc\ReferenceGuide
```

Obtaining All Information about SCSI Devices

The sample file "SCSILunName.java," demonstrates obtaining information (canonicalName, alternateName, among other properties) about physical LUNs that can then be used to correlate a given LUN name to all instances of that LUNs usage.

The Java sample (SCSILunName.java) demonstrates obtaining the complete set of SCSI LUN properties. The ScsiLun data object includes two new properties—alternateNames, standardInquiry—that can be used to correlate multiple LUN UUIDs to a single hardware device.

See the VI API Reference Guide for more information. The API Reference Guide is available in the VI SDK 2.5 package, in this path:

```
\SDK\doc\ReferenceGuide
```

Using HTTP for Cold Migration of Virtual Machine Files

The GetVMFiles.java and ColdMigration.java samples demonstrate downloading and uploading all files associated with a virtual machine (configuration, snapshot, suspend, log, and virtual disk files) to and from ESX Server or VirtualCenter using the HTTP protocol. The samples can be used to perform a cold-migration of all files (*.vmdk, *.vmx, and so on) associated with a specified virtual machine (VM).

Java Samples Summary

Table A-5 lists the Java package names and content for samples in this release. (Note that inner classes and additional supporting utility class files may not be listed in the table.)

Table A-5. Java samples listing (path starts from \SDK\samples\Axis\java\com\vmware)

Directory	Description	Java filename
apputils	Basic client-side utility classes that support logging, connecting to the service. Used by the other samples to handle client- application basics (connect, login, logout).	AppUtil.java
		ClientUtil.java
		Log.java
		OptionSpec.java
apputils\vim	Refactored client-side utilities that facilitate connecting to the server, authenticating, and other basics, using the vim WSDL. (The BaseClient.java and related files in prior SDK releases).	ServiceConnection.java
		ServiceUtil.java
		VmUtils.java
apputils\vim25	Client-side utilities that facilitate connecting to the server, authenticating, and other basics, using the vim25 WSDL.	ServiceConnection.java
		ServiceUtil.java
samples\alarms	Creates an Alarm to monitor the power state of the specified virtual machine.	VMPowerStateAlarm.java
samples\events	Retrieve and Format the latest event. Demonstrates Event formatting.	EventFormat.java
	Create an event history collector managed object and obtain a reference to its most recent page.	EventHistoryCollectorMonitor.java
	Create an event history collector for a single virtual machine and get its latest page.	VMEventHistoryCollectorMonitor.java

Table A-5. Java samples listing (path starts from \SDK\samples\Axis\java\com\vmware) (Continued)

Directory	Description	Java filename
samples\general	Obtains content of the specified service, starting with the root folder, and displays listing of all properties.	Browser.java
	Connect to specified service, login using specified credentials, logout.	Connect.java
	Creates Folder, Datacenter, and Cluster managed entities.	Create.java
	Deletes Folder, Datacenter, and Cluster managed entities.	Delete.java
	Displays update information from various virtual machines and hosts.	GetUpdates.java
	Demonstrates uses of the Licensing API.	LicenseManager.java
	Moves the specified Managed Entity,.	Move.java
	Demonstrates using the PropertyCollector managed object.	PropertyCollector.java
	Renames a Managed Entity	Rename.java
	Demonstrates using the SearchIndex managed object.	SearchIndex.java
	Connects to specified service, logs in (if appropriate, assuming the server is configured to support HTTPS only), retrieves ServiceContent managed object and then obtains inventory (managed entities), displays listing to console, and then logs out.	SimpleClient.java
	Displays details about currently running tasks.	TaskList.java
samples\host	Adds a Host VirtualNic to a PortGroup on a Virtual Switch.	AddVirtualNic.java
	Adds a Virtual Switch.	AddVirtualSwitch.java
	Adds a port group to a virtual switch.	AddVirtualSwitchPortGroup.java
	Removes a Host VirtualNic from a PortGroup on a Virtual Switch.	RemoveVirtualNic.java
	Removes the specified virtual switch.	RemoveVirtualSwitch.java
	Removes a port group from a virtual switch.	RemoveVirtualSwitchPortGroup.java
samples\httpfileaccess	Uses HTTP "put" to copy specified local fileset (configuration, snapshot, suspend, log, and virtual disk files comprising virtual machine) to a specified datacenter and datastore of a ESX Server. Also registers and reconfigures the virtual machine on the target ESX Server.	ColdMigration.java
	Uses HTTP "Get" to retrieve all files associated with a specified virtual machine (configuration, snapshot, suspend, log, and virtual disk files) and copies them to a specified local directory. (Use the ColdMigration.java sample to subsequently move the files to a different server.)	GetVMFiles.java

Table A-5. Java samples listing (path starts from \SDK\samples\Axis\java\com\vmware) (Continued)

Directory	Description	Java filename
samples\performance	Reads performance measurements from the past history.	History.java
	Prints all managed entity counters (virtual machine, resource pool) to a specified target file.	PrintCounters.java
	Reads performance measurements from the current time.	Realtime.java
	An ESX Top look-alike demonstrating VI Performance monitoring API. Note: This sample is non-functional at the present time.	VITop.java
	Demonstrates how to use the PerformanceManager managed object type.	VIUsage.java
samples\scheduling	Deletes a one time scheduled task named VMPowerOffTask created in the OneTimeScheduledTask sample to power off a virtual machine.	DeleteOneTimeScheduledTask.java
	Creates a one time scheduled task named VMPowerOffTask to power off a virtual machine.	OneTimeScheduledTask.java
	Creates a Weekly recurrence scheduled task to reboot the guest of a virtual machine.	WeeklyRecurrenceScheduledTask.java
samples\scsilun	Shows how to use the new alternateName property to maintain consistent mapping of LUN names.	SCSILunName.java
samples\version	Connects to server, obtains version information (using GetVersion), and then displays server name (for ESX Server 3.0.x systems), or displays boot duration, power state, and time zone for ESX Server 3.5 systems.	DisplayNewProperties.java
	Obtains version information from the WSDL available on ESX Server and VirtualCenter Server. Other samples in this sub-directory use this class to obtain version information.	GetVersion.java
	Accessing a new property available on ESX Server 3.5 or VirtualCenter 2.5.	GetVirtualDiskFiles.java
	Demonstrates exercising an operation available on ESX Server 3.5 or VirtualCenter 2.5.	PowerDownHostToStandBy.java
	Demonstrates using a new operation (in place of a deprecated operation).	QueryMemoryOverhead.java
samples\vm	Demonstrates cloning a virtual machine.	VMClone.java
	Create a virtual machine.	VmCreate.java
	Obtains a reference to a virtual machine and invokes various power operations on the reference, as specified on the command line options, such as snapshot, powering-off, and so on.	VmPowerOps.java
	Reconfigure a virtual machine.	VmReconfig.java
	Performs virtual machine snapshot operations.	VmSnapshot.java