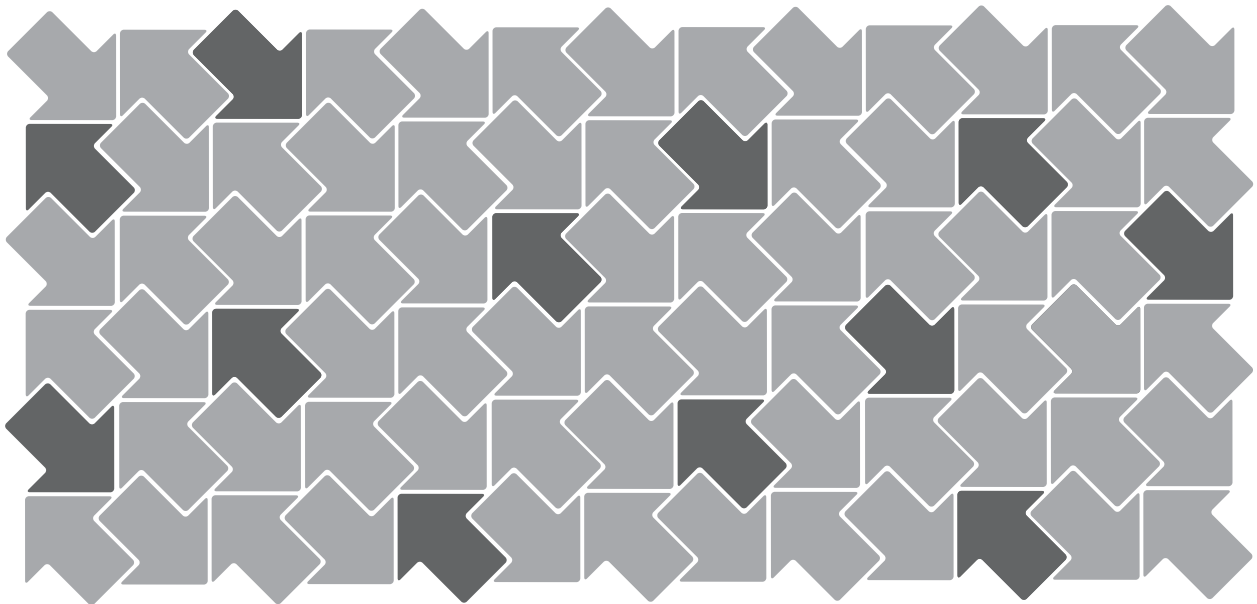


Developer's Setup Guide

VMware Infrastructure SDK 2.0.1



Developer's Setup Guide

Revision: 20070822

Item: VI-ENG-Q207-NEW

You can 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.

If you have comments about this documentation, submit your feedback to:

docfeedback@vmware.com

© 2007 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, 6,795,966, 6,880,022, 6,944,699, 6,961,806, 6,961,941, 7,069,413, 7,082,598, 7,089,377, 7,111,086, 7,111,145, 7,117,481, 7,149,843, 7,155,558; 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. All other marks and names mentioned herein may be trademarks of their respective companies.

VMware, Inc.

3401 Hillview Ave.

Palo Alto, CA 94304

www.vmware.com

Contents

About This Book	3
1 Introducing the VMware Infrastructure SDK	5
What is the VMware Infrastructure API?	5
Knowledge Required for Using the VI SDK	5
Programming Languages Supported by the VI SDK	6
Types of Applications That Can Be Built Using the VI SDK	6
What's in the VI SDK Package?	7
VI SDK Versions and VMware Infrastructure 3 Product Compatibility	9
2 Server-Certificate Requirements and Preliminary Setup	11
Obtaining Server Certificates	11
Obtaining Certificates by Using Microsoft Internet Explorer's Certificate Cache	12
Obtaining Certificates by Connecting Directly to Server Systems	13
Modifying the Server Configuration to Support HTTP	14
Modifying a VirtualCenter Server Configuration File	15
Modifying an ESX Server Configuration File	16
3 Setting Up for Java Development	19
Requirements	19
Setup Instructions for Java Development	20
Setting Environment Variables	20
Importing Server-Certificates into the Java Keystore	21
Generating Stubs and Compiling Classes	22
Running the SimpleClient Sample Application to Validate Setup	23
Troubleshooting Setup Issues	24
4 Setting Up for Microsoft C# Development	27
Requirements	27
Setup Instructions for C# Development	28
Running the Microsoft .NET (C#) Version of SimpleClient	30
Troubleshooting Setup Issues	31
A Reference	33
Batch File and Shell Script Inventory	33
Java Development	33
Microsoft .NET Development	33
Java Samples	34
Microsoft .NET (C#) Samples	36
Index	39

About This Book

This book, the *Developer's Setup Guide*, is a completely new book (as of August 23, 2007) for the VMware Infrastructure (VI) SDK 2.0.1: It provides information about setting up your environment for using the VI SDK 2.0.1 to develop client applications. To setup your development environment for working with the VI SDK 2.0.1, use this book instead of any setup information contained in the Getting Started Guide, Programming Guide, and various readme files (for VI SDK 2.0.1).

VMware provides several different SDK products, each intended for different developer communities and target platforms. This guide is intended for developers who are creating applications aimed at managing VMware virtual infrastructure through ESX Server and VirtualCenter Server systems.

Revision History

This guide is new as of August 23, 2007. [Table 1](#) describes the changes in each version of this guide.

Table 1. Revision History

Revision	Description
200709212	Minor edit to clarify that use of provided vim.jar is for test purposes only. Fixed several broken cross-reference links.
20070823	Initial publication of the Developer's Setup Guide. This is a new book that replaces installation and setup information in the VI SDK 2.0.1 Getting Started Guide, Programming Guide, and various readme files.

Intended Audience

This book is intended for anyone who wants to develop applications using the VI SDK. VI SDK developers typically include software developers creating virtual infrastructure management applications using Java or C# (in the Microsoft .NET environment) targeting the Web-services based API available on ESX Server and VirtualCenter Server systems.

Terminology Used in this Guide

To simplify the discussion, this guide uses these terms:

- **Target server**—The VirtualCenter Management Server or ESX Server systems that are the targets of your client-side code.
- **Development workstation**—The Linux or Microsoft Windows machine that is configured with the Web-services client-side libraries, development environment, and the SDK download.

Document Feedback

VMware welcomes your suggestions for improving our documentation. If you have comments, send your feedback to:

docfeedback@vmware.com

Conventions

Table 2 illustrates the typographic conventions used in this guide.

Table 2. Conventions Used in This Guide

Style	Elements
Blue (online only)	Links, cross-references, and email addresses
Black boldface	User interface elements such as button names and menu items
Monospace	Commands, filenames, directories, and paths
Monospace bold	User input
<i>Italic</i>	Document titles, glossary terms, and occasional emphasis
<Name>	Variable and parameter names

Technical Support and Education Resources

The following sections describe the technical support resources available to you.

Self-Service Support

Use the VMware Technology Network (VMTN) for self-help tools and technical information:

<http://www.vmtn.net>

Here are some other links for self-service support:

- Product information – <http://www.vmware.com/products/>
- Developer Center – <http://www.vmware.com/communities/content/developer>
- Documentation – <http://www.vmware.com/support/pubs>
- Knowledge Base – <http://kb.vmware.com>
- Discussion forums – <http://www.vmware.com/community>
- User groups – <http://www.vmware.com/communities/content/vmug/index.html>

Online Support

You can submit questions or post comments to the VMware Management APIs (VI SDK, VI Perl, CIM SDK) forum, which is monitored by VMware technical support and product teams. You can access the forum at:

<http://www.vmware.com/community/forum.jspa?forumID=393>

Support Offerings

Find out how VMware support offerings can help meet your business needs. Go to <http://www.vmware.com/support/services>.

VMware Education Services

VMware courses offer extensive hands-on labs, case study examples, and course materials designed to be used as on-the-job reference tools. For more information about VMware Education Services, go to <http://mylearn1.vmware.com/mgrreg/index.cfm>.

Introducing the VMware Infrastructure SDK

1

The VMware Infrastructure SDK (VI SDK) facilitates development of client applications that target the VI API.

VMware provides several different SDK products, each intended for different developer communities and target platforms. The VI SDK is for developers who want to create client applications that can manage, monitor, and maintain VMware virtual infrastructure (deployed on ESX Server and VirtualCenter Server systems).

This chapter provides an overview of VI SDK packaging, requirements, and functionality. It includes these topics:

- [What is the VMware Infrastructure API?](#)
- [Knowledge Required for Using the VI SDK](#)
- [Programming Languages Supported by the VI SDK](#)
- [Types of Applications That Can Be Built Using the VI SDK](#)
- [What's in the VI SDK Package?](#)
- [VI SDK Versions and VMware Infrastructure 3 Product Compatibility](#)

What is the VMware Infrastructure API?

The VMware Infrastructure API (VI API) provides a complete set of language-neutral interfaces to the VMware virtual infrastructure management (VIM) framework. In much the same way that JMX (Java Management Extension) provides an infrastructure for instrumenting, managing, and monitoring Java applications, VIM provides an infrastructure for basically doing the same for VMware Infrastructure 3 components (for example, virtual machines, host systems) and subsystems (such as performance manager). VIM is accessed by external clients using the VI API.

VI API is implemented as industry-standard Web services, hosted on VirtualCenter Server and ESX Server systems. The VI API complies with the Web Services Interoperability Organization (WS-I) Basic Profile 1.0, which includes XML Schema 1.0, SOAP 1.1, WSDL 1.1. For more information about the WS-I Basic Profile 1.0, see:

<http://www.ws-i.org/Profiles/BasicProfile-1.0-2004-04-16.html>

The Web service provides all the operations necessary, including life-cycle operations, to monitor and manage virtual infrastructure components—compute resources, virtual machines, networks, storage, and the like.

Knowledge Required for Using the VI SDK

To be successful with the VI SDK, you should be familiar with basic Web services concepts, in general, and understand specific programming and debugging processes and techniques. Some concepts you most need to understand are these:

- Web services technology provides operations (same basic concept as “methods” in other programming languages). Using the VI SDK and the programming language of your choice, you can create client applications that invoke these operations to perform the full range of server-side management and monitoring tasks.
- The Web services API is defined in a WSDL (web services description language) file. The WSDL file is used by client-side Web-services utilities to create proxy code (stubs) that client applications use to interact with the server.
- Client applications invoke operations by sending SOAP (simple object access protocol)-formatted messages. (SOAP is an XML format and is programming-language neutral. One of the jobs of the client-side Web services tools is formatting (transparent to you, the developer) the SOAP messages from the programming language that you use.
- Communications between client and server occur over HTTP or HTTPS (secure HTTP, which uses SSL to encrypt client-server communications). The default is HTTPS, but the Web server (on ESX Server and VirtualCenter Server) can be configured to support HTTP. (See “[Modifying the Server Configuration to Support HTTP](#)” on page 14 for details.)

In addition, you should be familiar with the operation of VMware VirtualCenter and VMware ESX Server—the targets for your coding efforts. For more information, including an overview of basic technology and product concepts, see the VMware Infrastructure 3 Quick Start Guide:

http://www.vmware.com/support/pubs/vi_pubs.html

Programming Languages Supported by the VI SDK

Technically speaking, since the VI API is based on Web services, you can use any programming or scripting language that provides utilities for generating client-side stubs (proxy code) from Web-services WSDL files. However, to facilitate your programming efforts, the VI SDK includes sample applications in both Java and C#, so these are recommended. (See [What's in the VI SDK Package?](#) for additional packaging details, and for some caveats regarding Java samples and specific versions of the JDK and Axis.)

Table 1-1. Language and tools matrix for client application development

	Java	C#
Development environment or framework	J2SE 5.0 (aka, J2SE 1.5—J2SE 1.5_0_08 or subsequent version recommended) J2SE 1.4.x	Microsoft Visual Studio 2005 (recommended) Microsoft Visual Studio 2003 Microsoft Visual C#
Web-services client development toolset	Apache Axis 1.2.1 (recommended) Apache Axis 1.4	Microsoft .NET Framework 2.0 (recommended) Microsoft .NET 1.1

Note that Perl scripters can access the VI API through the VI Perl Toolkit. Designed to minimize the client-side development overhead (mapping datatypes and generating client proxy code, for example) associated with distributed-object computing, the VI Perl Toolkit includes a runtime component that handles language bindings, remote instantiation, and other Web-services-client programming infrastructure issues.

You can learn more about the VI Perl Toolkit at:

<http://www.vmware.com/support/developer/viperltoolkit>

Types of Applications That Can Be Built Using the VI SDK

The types of applications that you can develop using the VI SDK include system administration, provisioning, and monitoring applications for VMware Infrastructure 3 systems. For example, VMware Infrastructure Client (VI Client) application and VMware Virtual Infrastructure Web Access interface have both been created using VI SDK components. The VI Client is a traditional Windows client application; Web Access is a browser plug-in that is available via the Web server port on ESX Server and VirtualCenter Management Server systems.

With the VI SDK, you can go beyond the capabilities of the VI Client and Web Access tool, by automating or otherwise streamlining the various administration, provisioning, or monitoring tasks associated with virtual infrastructure management and operations.

For example, you can create a client application that creates multiple virtual machines across several different ESX Server systems automatically, perhaps when initiated from a Web-server submission form.

Here are some examples of the operational tasks that you can automate using the VI API:

- Create, configure, power-cycle, or suspend virtual machines explicitly, or by using cloning or templates to facilitate faster provisioning.
- Create, configure, and manage virtual devices, such as virtual CD-DVD drives, virtual network interface cards, virtual switches, and other components.
- Connect, power-cycle, and disconnect host systems (ESX Server) and VirtualCenter Management Server.
- Capture the state of a virtual machine (to a snapshot) and restore the state of a virtual machine from a snapshot.
- Gather statistics about host system and virtual machine performance.
- Manage events generated by the server, such as those emitted by alarms set for specific thresholds.
- Move virtual machines between hosts automatically.
- Manage load balancing and failover through the distributed resource scheduler (VMware DRS) and high availability (VMware HA) sub-systems.

This list is not comprehensive. Also, note that some of the operational tasks pertain to the service as a whole, not specific hosts or virtual machines. For example, load balancing can be a service-wide function rather than per-host or per-virtual machine functions.

What's in the VI SDK Package?

Available for download from the VMware Web site (<http://www.vmware.com/download/sdk/index.html>), the VI SDK is a zipfile (`vi-sdk-2.0.1-32042.zip`) that includes:

- Sample code demonstrating common use cases associated with programmatically managing a virtual infrastructure. The sample code includes compiled and ready-to-run Java class files, as well as Java and C# source code files. The compiled Java samples are located in the following sub-directory of the SDK download (unpacked):

```
SDK\samples_2_0\Axis\java\com\vmware\vimsample.classes
```

NOTE The pre-compiled Java samples have been compiled using the JDK 1.5 and use Apache Axis 1.2.1 libraries. The samples work only with these specific versions of Java JDK and Axis. Furthermore, JDK 1.5.0_08 (or subsequent version) is recommended, to avoid a performance issue.

- WSDL files that define the API available on an ESX Server and VirtualCenter Server Web service, and pre-compiled client-side libraries (to use for testing purposes) that have been generated from the WSDL. For C# developers, the Microsoft Visual Studio project files (.sln) have been included.
- Batch files and shell scripts to automate the build process for Java client applications.
- Complete API reference documentation (VI API ReferenceGuide) that provides language-neutral descriptive information (object type definitions, properties, and method signatures, for example) about the VMware Infrastructure API (VI API) and the server-side object model.
- Technical publications, including this Developer's Setup Guide and a Programming Guide, that provide conceptual and prescriptive information about how to develop client applications using the VI SDK.

When you unpack the VI SDK download, you'll see something like the simple directory structure shown in [Figure 1-1](#).

See [Appendix A, “Reference,”](#) on page 33 for an inventory of sample applications for C# and Java, and reference information about the various batch files and shell scripts.

VI SDK Versions and VMware Infrastructure 3 Product Compatibility

VMware has released several VI SDK products to support various versions of the VMware Infrastructure product family, as shown in [Table 1-2](#).

Table 1-2. VI SDK versions and VMware Infrastructure 3 version compatibility

VI SDK	Supported target server version	Usage note or caveat
VI SDK 2.0.1	ESX Server 3.0.x	Supports vim2 API.
	ESX Server 2.x	Support provided through VirtualCenter Server 2.x.
	VirtualCenter Management Server 2.x	Supports vim2 API.
VI SDK 1.x	ESX Server 3.0.x	Support provided through VirtualCenter Server 2.x.
	ESX Server 2.x	Support provided through VirtualCenter Server 2.x.
	VirtualCenter Management Server 2.0.x	Supports vim2 API. Must enable 1.x compatibility mode.
	VirtualCenter Server 1.4.x	Supports vim1 API.

Server-Certificate Requirements and Preliminary Setup

2

The VMware Infrastructure API is available as a secure Web service running on VirtualCenter Management Server system (2.x and later) and on ESX Server (3.x or later). Before you can get started with the VI SDK, you must have access to one of these two VMware products.

No additional server-side installation or configuration is required to enable client applications to access the API via these Web services. However, depending on the specific configuration of the server that you will be using for development purposes, you must perform one of these two preliminary tasks:

- Obtain the server-certificates for the ESX Server or VirtualCenter Management Server that you plan on using during development, and import them into the local certificate store of the development workstation; or
- Modify the default server configuration, so that it supports non-SSL (regular HTTP) communications, in which case you do not need to import the server certificates on the client development workstation.



CAUTION VMware recommends that secure HTTP (HTTPS; the default configuration) be used for production deployments. Modifying the server configuration to support HTTP access to the VI API is recommended for **test or development environments only, not for production deployments.**

This chapter includes information about both alternatives. It includes these topics:

- [Obtaining Server Certificates](#)
 - [Obtaining Certificates by Using Microsoft Internet Explorer's Certificate Cache](#)
 - [Obtaining Certificates by Connecting Directly to Server Systems](#)
- [Modifying the Server Configuration to Support HTTP](#)

Obtaining Server Certificates

VMware products use standard X.509 version 3 (X.509v3) certificates to encrypt session information sent over SSL (secure sockets layer) connections between server and client systems. These certificates are created automatically during the process of installing VMware products, including ESX Server and VirtualCenter Server systems. (Since these default certificates are digitally signed with the name of the host system, they are sometimes referred to as “self-signed certificates.”) You must obtain the server certificate from each server that you plan to target with your client application (the “target servers”).

For example, if you are creating a client application that will run against the VirtualCenter Server and an ESX Server directly (in standalone mode), you must obtain both the VirtualCenter Management Server certificate and the ESX Server certificate. On the other hand, if your application will run solely against the VirtualCenter Management Server (that may be managing any number of ESX Server systems), you must obtain the certificate from the VirtualCenter Management Server only.

You can obtain the certificates in one of two general ways:

- Developers working on the Microsoft Windows platform can use the certificate-handling capabilities of the Internet Explorer browser (from the development workstation) to quickly browse to each ESX Server or VirtualCenter Management Server and accept the certificate into the local cache (as detailed in [“Obtaining Certificates by Using Microsoft Internet Explorer’s Certificate Cache,”](#) below).
- Developers with access privileges on the target server systems can use a secure shell client utility (SCP, WinSCP, or SSH) to connect directly to the ESX Server or VirtualCenter Management Server and copy the certificates directly from the server to the development platform. See [“Obtaining Certificates by Connecting Directly to Server Systems”](#) on page 13 for details.

Obtaining Certificates by Using Microsoft Internet Explorer’s Certificate Cache

This approach works with Microsoft Internet Explorer browser only. Mozilla Firefox handles certificates differently, so these instructions can not be applied to the Firefox (or other, non-Internet Explorer) browsers, nor to non-Windows platforms.

To obtain server certificates using Microsoft Internet Explorer (for Windows development platform)

- 1 From the development workstation, open Internet Explorer browser.
- 2 Navigate to the ESX Server or VirtualCenter Management Server web server using the HTTPS protocol:

https://servername

A Security Alert message displays a warning regarding the certificate’s certifying authority.

NOTE Message text will vary, depending on the version of Microsoft Internet Explorer that you use. The warnings are raised because the default certificates are self-signed. If your site has replaced the default certificates, you will likely not see the warning messages. However, you will still need to obtain the certificates and install on the local development workstation as discussed below.

- 3 Click **View Certificate** to open the Certificate properties page.
- 4 Click **Install Certificate** to launch the Certificate Import Wizard. Keep the default setting (“Automatically select the certificate store based on the type certificate”) and click **Next** to continue.
- 5 Click **Next** to continue installing the certificate.
- 6 Click **Finish**. A Security Warning message displays regarding the certificate’s certifying authority.
- 7 Click **Yes** to continue with the certificate installation. A Certificate Import Wizard “success” message displays.
- 8 Click **OK** to dismiss the success message. The Certificate properties page becomes active again.
- 9 Click **OK** in the Certificate dialog box to continue to the server. The initial Security Alert message presented in step 2 above becomes active again (returns to focus in the window).

Click **Yes** in the Security Alert message to continue with the original HTTPS request for the server (back at step 2, where all this began). The server (VMware ESX Server 3, VMware VirtualCenter 2) Welcome page displays.

The Certificate has now been installed in Internet Explorer’s certificate cache.

- 10 Repeat the process for each ESX Server and VirtualCenter Management Server that you will be using with the VI SDK.

Once you’ve obtained all the certificates you need, one for each target server, next steps depend on the programming language you’ll be using:

- For **C# (Microsoft .NET) developers**, at this point, preliminary setup tasks are complete: the .NET infrastructure uses the certificates you’ve imported. You can continue setting up your development workstation by following the instructions in [Chapter 4, “Setting Up for Microsoft C# Development,”](#) on page 27.

- For **Java developers**, you must export the certificates from the Internet Explorer cache to a local directory. Minimize the Internet Explorer browser window (or simply move it aside for a moment), and export the certificates as detailed in the next several steps.

To export the cached certificates to a local directory (for Java development on Windows)

- 1 Create a directory for the certificates, using the name set in the various batch files for the VI SDK:
C:\VMware-Certs
- 2 From the Internet Explorer **Tools** menu, select **Internet Options** to open the Internet Options properties page.
- 3 Click the **Content** tab to activate content advisor, certificates store on the properties page.
- 4 Click **Certificates** to open the Certificate manager.
- 5 Click the **Trusted Root Certificate Authorities** tab to display the list of trusted certificates. This list should contain a certificate for each of the target servers selected in steps 2 through 8, above.
- 6 Scroll through the list of certificates to find the certificates. For ESX Server systems, the certificate name matches the DNS name of the server. For VirtualCenter Server systems, the certificate name is VMware.
- 7 For each target server:
 - a Click the certificate to select it.
 - b Click **Export . . .** to launch the Certificate Export Wizard.
 - c Click **Next** to continue. The Export File Format dialog displays.
 - d Keep the defaults (“DER encoded binary X.509 (.CER)”) and click **Next** to continue. The File To Export dialog displays, enabling you to enter a unique name for the certificate.
 - e Choose a filename and enter it, along with the complete path to the directory created in step 9:
C:\VMware-Certs\servername.cer

NOTE If you don’t enter the complete path, the certificate gets stored in your “Documents and Settings” folder. On Windows development workstations, the batch files included with the VI SDK have been defined to use the c:\VMware-Certs directory.

- f Click **Next** to continue with the export. A Completing the Certificate Export Wizard page displays, summarizing the information about the certificate.
 - g Click **Finish** to complete the export. A Certificate Export Wizard “success” message displays.
 - h Click **OK** to dismiss the success message.
- 8 When you have exported all certificates for the servers you want to target, click **Close** to exit the certificates export dialog.
 - 9 Click **Cancel** to close the Internet Options properties page.

Continue setting up your Java development environment by following the steps in [“Setting Up for Java Development.”](#)

Obtaining Certificates by Connecting Directly to Server Systems

This approach can be used by developers using Linux for development workstation, or anyone who has appropriate privileges to directly connect to the target server.

To obtain server certificates using secure shell client application (for Linux development platform)

These instructions require administrative privileges on the VirtualCenter Management Server, and assume that you can access the necessary sub-directory (see [Table 2-1](#)).

- 1 From the development workstation, create a directory in which to store certificates of ESX Servers or VirtualCenter Management Servers that you'll be using during development:

```
~\vmware-certs\
```

- 2 Connect to the ESX Server using an SSL client from the development workstation. (Remember that remote connections to the ESX Server service console as root are effectively disabled: you must connect as another user with privileges on the server, to obtain the certificate.)

The server certificate filenames and locations on various versions of the ESX Server and VirtualCenter Management Server are listed in [Table 2-2](#).

Table 2-1. Directory Locations and Certificate Filenames

Server	Directory Location for Certificate	Certificate
ESX Server 2.x (MUI)	/etc/vmware-mui/ssl/	mui.crt
ESX Server 3.x, ESX Server 2.x, GSX Server 3.x	/etc/vmware/ssl/	ru1.crt
VirtualCenter Server 1.x, VirtualCenter Server 2.x	C:\Documents and Settings\All Users\Application Data\VMware\VMware VirtualCenter\SSL\	ru1.crt

- 3 Copy the certificates from the server to the certificate sub-directory of the development workstation, using a unique filename for the certificate (assuming you are copying multiple default certificates from multiple ESX Server systems, for example).
- 4 Import the server-certificate into the certificate store following the specific instructions for your programming language (Java, C#). The details are covered in the "[Setting Up for Java Development](#)," and in "[Setting Up for Microsoft C# Development](#)," respectively.

Modifying the Server Configuration to Support HTTP

Both ESX Server 3.x and VirtualCenter Management Server 2.x support the VI API through their respective Web services (SOAP) engines. By default, these Web services run on port 443, as a secure Web service that can be accessed using SSL over HTTP (HTTPS), as in:

```
https://yourservername.fqdn.com/sdk
```

For a development environment, you may prefer to configure target servers (following the instructions in this section) to support regular (non-SSL) HTTP, to simplify your client-side setup, development, and testing. The configuration files are standard web server configuration files that you (or your system administrator) can manually edit using any text editor to:

- Add a deployment descriptor tag for the server and its port to the HTTP section of the configuration file
- Remove the redirection tag for the server from the HTTPS section of the configuration file

Table 2-2. Configuration Filenames and Locations

Server platform	Path	Filename
VirtualCenter Management Server (2.x and later)	c:\Documents and Settings\All Users\Application Data\VMware\VMware VirtualCenter	vpzd.cfg
ESX Server (3.x and later)	/etc/vmware/hostd/	config.xml

The next two subsections provide detailed instructions for making these changes on VirtualCenter and ESX Server, respectively.

Modifying a VirtualCenter Server Configuration File

The VirtualCenter Management Server configuration file is located several directories deep in the Windows Documents and Settings folder, in:

```
c:\Documents and Settings\All Users\Application Data\VMware\VMware VirtualCenter\vpzd.cfg
```

These instructions require access to the VirtualCenter Management Server, and that you can access the file and perform the necessary edit.

To edit a VirtualCenter Management Server configuration file (vpzd.cfg)

- 1 Logon to the Windows machine running VirtualCenter Management Server using the same credentials as used to install VirtualCenter Management Server.
- 2 Navigate to the directory where vpzd.cfg is located:
cd c:\Documents and Settings\All Users\Application Data\VMware\VMware VirtualCenter
- 3 Use a text editor, such as WordPad, to open the configuration file. (Note that the file may contain commented instructions about how to modify to support HTTP. If it does, follow the instructions in the file, rather than these.)
- 4 Search through the vpzd.cfg file to locate the <proxyDatabase> tag within the <http> tag.

```

vpzd.cfg
<!-- <eventMap> C:\wsixEventMap.xml </eventMap> -->
<!-- <wsix> -->
<vpzd>
  <proxySvc>
    <http>
      <!-- Web server database for the http proxy -->
      <!-- port = -1: Built-in webservice -->
      <!-- port = -2: Built-in soap port -->
      <!-- port = -3: Built-in mob port -->
      <!-- registryKey: Read port from Windows registry -->
      <proxyDatabase>
        <server id="0">
          <namespace / </namespace>
          <host localhost />
          <port -1 />
        </server>
        <redirect id="2"/>/ui</redirect>
        <redirect id="3"/>/mob</redirect>
        <!-- uncomment the entries below and remove redirect entries (above) -->
        <!-- to allow non-SSL access to the sdk and mob -->
        <server id="1"> -->
          <namespace /sdk />
          <host localhost />
          <port -2 />
        </server> -->
        <!-- <server id="3"> -->
        <!-- <namespace /mob /> -->
        <!-- <host localhost /> -->
        <!-- <port 8087 /> -->
        <!-- </server> -->
      </proxyDatabase>
    </http>
    <https>
      <!-- Web server database for the https proxy -->
      <proxyDatabase>
        <server id="0">
          <namespace / </namespace>
          <host localhost />
          <port -1 />
        </server>
        <server id="1">
          <namespace /sdk />
          <host localhost />
          *****

```

- 5 Within the <proxyDatabase> tag, add the following text (or remove the comment code to enable the setting, if the deployment descriptor tags already exist in the file):

```

<server id="1">
  <namespace /sdk />
  <host localhost />
  <port -2 />
</server>

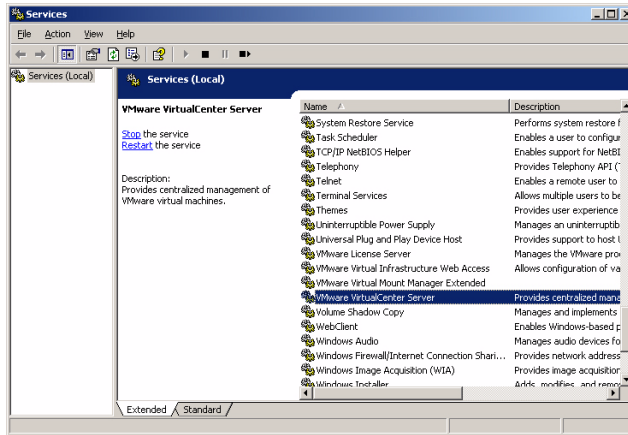
```

- 6 In the <proxyDatabase> tag (within the <http> definition), delete the redirection tag associated with the /sdk, which may look like the following:

```
<redirect id="2"> /sdk </redirect>
```

- 7 Save changes and close the file.
- 8 Restart the VMware VirtualCenter Server service using the Services control panel:

- a From the Windows Start menu, select Settings-->Control Panel-->Administrative Tools-->Services. The Services control panel opens, displaying the full list of services installed on the server.
- b Scroll through the list of services to find the VMware VirtualCenter Server.



- c Click the VMware VirtualCenter Server service to highlight it, and then click **Restart**. The service restarts and the new server settings take effect.

The server should now be accessible via HTTP connection. To test, open a browser and connect to the VirtualCenter Server using HTTP.

Modifying an ESX Server Configuration File

The ESX Server configuration file is located in:

```
/etc/vmware/hostd/config.xml
```

The instructions below require administrative (root) privileges on the ESX Server system. Note that, by default, remote connections (SSH, for example) to ESX Server service console are disabled. Typically, another user account is used to connect to the server and then **su - root** to perform tasks at the service console.

To edit an ESX Server configuration file (config.xml)

- 1 Connect to the ESX Server and navigate to the /etc/vmware/hostd directory.
- 2 Use a text editor, such as vi, to open the config.xml file. You'll see several XML element tags defining the Web server subsystems. The changes you need to make are all in the <proxySvc> element.
- 3 Find the section of the file in the <proxySvc> element, under the <https> tag. In this section, you'll see tags identifying the <port> and the <proxyDatabase>.
- 4 Under <proxyDatabase>, locate the tag sequence for the VI API server port (/sdk). The tag sequence may look like this example:

```
<server id="1">
  <namespace> /sdk </namespace>
  <host> localhost </host>
  <port> 8085 </port>
</server>
```

- 5 Copy the entire set of tags from `<server id="n">` through its matching closing tag `</server>`, and paste within the `<proxyDatabase>` element (in the `<http>` element section), as shown in this example:

NOTE This screenshot shows an example `config.xml` of a server that has been configured to support HTTP.

- 6 In the `<proxyDatabase>` tag (within the `<http>` definition), delete the redirection tag associated with the `/sdk`, which may look like the following:


```
<redirect id="2"> /sdk </redirect>
```
- 7 Return to the `<proxyDatabase>` tag within the `<https>` definition and delete the content that you copied in step 4 and 5 above.
- 8 Save changes and close the file.
- 9 Restart the Web service:

```
service mgmt-vmware restart
```

The server should now be accessible via HTTP connection.

VMware, Inc.

21-Sept-2007

17

Setting Up for Java Development

To run any of the sample applications, or to start creating, compiling, and testing your own client applications, you'll need to configure your development workstation with the necessary Web-services-client tools and perform several other setup tasks, depending on the specifics of your environment. This chapter provides detailed setup information for developers using Java. It includes these topics:

- [Requirements](#)
- [Setup Instructions for Java Development](#)
- [Running the SimpleClient Sample Application to Validate Setup](#)
- [Troubleshooting Setup Issues](#)

Requirements

Developing Java Web-services client applications using the VMware Infrastructure SDK requires the Java SDK and a Java Web services development toolset, specifically:

- **Java 2, Standard Edition, version 5.0** (J2SE 1.5.x) or J2SE 1.4.x. **VMware recommends using J2SE 1.5.0_08 (or later)**, for two reasons:
 - Improved performance—Using Apache Axis client-libraries with versions of the JDK prior to JDK 1.5_0_08 yields slow performance (due to a socket-creation issue).
 - Samples support—Some of the samples use features not supported in versions of the JDK 5 (JDK 1.5).
- **Apache Axis 1.2.1**—Axis is an open source project of the Apache Web services project. It's a SOAP ("Simple Object Access Protocol") implementation that can be deployed to a Tomcat server. The client-side components include various XML processing and other libraries needed for Web-services client development. You can obtain Axis from the Apache Web-services project site:

<http://ws.apache.org/axis/>

Other client-side tools and libraries, such as IBM's WebSphere and several open source implementations, can be used with the VI SDK. However, only the Axis client libraries have been tested with this Developer's Setup Guide. Also, note these constraints regarding Axis versions and the included sample applications:

- The **Apache Axis 1.2.1** libraries were used to generate the client proxy code (stub classes) from which the VI SDK 2.0.1 precompiled `vim.jar` and Java sample applications (`vimsamples.jar`) were made. These samples have been compiled using J2SE 5 (JDK 1.5.0) and can be run using Java 1.5. Several of the samples use features in Java 1.5.0 and will not compile using versions of the JDK prior to 1.5.
- **Apache Axis 1.4** is supported for use with VMware Infrastructure 3. However, the samples provided in the SDK 2.0.1 are compatible with stubs generated using Axis 1.2.1 only.
- Because of a difference in some method signatures between stubs generated by Axis 1.2.1 and Axis 1.4, if the code you write must work with *either* version of Axis, you cannot use any stub-class constructors that take arguments. Instead, you should initialize values as needed using `set()` methods of the stub classes.

Setup Instructions for Java Development

Specific setup instructions will vary, depending on whether your development workstation already meets some or all of the requirements, and whether you plan to use the provided samples. The VI SDK package includes a `vim.jar` file that was generated using Apache Axis 1.2.1 libraries and compiled using Java JDK 1.5—you can use this `vim.jar`, as is, without generating new stubs and re-compiling—if you are using these same versions of the JDK and Axis.

To setup a development workstation to use Java

- 1 Create directories for each of the components that you will need to install (assuming that neither JDK nor Axis client is installed). Do not use spaces in the directory names, to avoid issues with some of the included SDK batch and script files. [Table 3-1](#) lists some recommended naming conventions:

Table 3-1. Recommended directory structures

	Windows	Linux
VI SDK	<code>c:\devprojects\visdk201</code>	<code>~\apps\visdk</code>
Apache Axis	<code>c:\apache\axis</code>	<code>~\apps\apache\axis</code>
Java 2, Standard Edition (J2SE)	<code>c:\java\jdk1.5.0_mn</code>	<code>~\apps\java\jdk1.5.0_mn</code>
	<code>c:\java\jre1.5.0_mn</code>	<code>~\apps\java\jre1.5.0_mn</code>

- 2 Install the Java 2 Platform, Standard Edition (J2SE) 5.0. VMware recommends using JDK 1.5_0_08 or later due to improved performance (for Java sockets). You can obtain the J2SE from:

<http://java.sun.com/javase/downloads/previous.jsp>
- 3 Obtain the Apache Axis 1.2.1 client-side Web services libraries from the Apache Web services website, at:

<http://ws.apache.org/axis/java/releases.html>
- 4 Obtain the VMware Infrastructure SDK (VI SDK) package from VMware Web site:

<http://www.vmware.com/download/sdk/>
- 5 Unpack the various components into sub-directories created in step 1 above, using the provided installer if appropriate. (The J2SE uses an installation wizard; the Axis zipfile and the SDK simply unpack into a selected target sub-directory. Be sure to unpack with “Use folder names” selected, to maintain the organizational structure.)
- 6 Configure environment settings as detailed “[Setting Environment Variables](#),” below.
- 7 Import server-certificates (assuming that the target servers are configured for HTTPS) and create a `vmware.store` (using the Java keytool utility). See “[Importing Server-Certificates into the Java Keystore](#)” on page 21 for details. At runtime, when you execute the Java bytecode, you must pass the keystore file name to the Java runtime as a property, to support the SSL connection.
- 8 Use the `build.bat` (or `build.sh`, on Linux) to generate new stubs and compile into `vim.jar`. The build scripts perform all necessary tasks for you. (If you are using Axis 1.2.1 and JDK 1.5, you do not need to rebuild—simply add `vim.jar` to your system classpath.) See “[Generating Stubs and Compiling Classes](#)” on page 22 for details.
- 9 Run the Java version of SimpleClient to test your setup. (See [Running the SimpleClient Sample Application to Validate Setup](#) for details). Note that you may have to regenerate client-side stubs and then recompile to create new class files—the precompiled samples and the provided `vim.jar` have been created using JDK 5 and Axis 1.2.1.

Setting Environment Variables

You must make changes to existing system CLASSPATH and PATH environment variable settings, to include the Axis libraries and the J2SE libraries. (If these variables do not exist already on your development workstation, you should create them.)

In addition, if you intend to use any of the provided batch files or shell scripts, you must create environment variables for `AXISHOME`, `JAVAHOME`, and `SDKHOME`—the batch files and shell scripts refer to these environment variables and will fail if these variables do not exist, or are set incorrectly.

[Table 3-2](#) lists the environment variable names, directories, and filenames that must be specifically set. Refer to [Table 3-2](#) for details about each environment variable as you follows these instructions.

- 1 Create environment variables for `AXISHOME`, `JAVAHOME`, and `SDKHOME`, as defined in [Table 3-2](#).
 - For Microsoft Windows development workstations, you should set System (rather than User) environment variables.
 - For Linux development workstations, you should export the variables in your user profile.
- 2 To the System `CLASSPATH`, add the complete path to all Axis client-side library JARs and additional libraries provided with the SDK specified in [Table 3-2](#).
- 3 To the System `PATH` environment variable, add the path to the Axis and Java runtime binaries, as specified in [Table 3-2](#).

Table 3-2. Environment variable names and required JAR and other files

Variable name	Setting should include...
<code>AXISHOME</code>	Complete path to the Apache Axis installation top-level directory. For example: C:\apache\axis1.2.1
<code>CLASSPATH</code>	Complete paths to specific JAR files and other libraries required by Java and Axis tools. Add these specific JAR files to the classpath (assumes you've setup <code>AXISHOME</code> , <code>JAVAHOME</code> , and <code>SDKHOME</code>): %AXISHOME%\lib\axis.jar %AXISHOME%\lib\axis-ant.jar %AXISHOME%\lib\commons-discovery-0.2.jar %AXISHOME%\lib\commons-logging-1.0.4.jar %AXISHOME%\lib\jaxrpc.jar %AXISHOME%\lib\log4j-1.2.8.jar %AXISHOME%\lib\saa.jar %AXISHOME%\lib\wsdl4j-1.5.1.jar %JAVAHOME%\lib\tools.jar %SDKHOME%\samples_2_0\Axis\java\vim.jar %SDKHOME%\samples_2_0\Axis\java\lib\activation.jar %SDKHOME%\samples_2_0\Axis\java\lib\mailapi.jar
<code>JAVAHOME</code>	Paths to the binary root directories for both the Java JDK and the Java runtime (JRE). For example: C:\jdk1.5_0_08\bin C:\jre1.5_0_08\bin
<code>PATH</code>	Add the path to the Java and the Axis binary client tools to the system path. Assuming you setup the <code>AXISHOME</code> and <code>JAVAHOME</code> variables, you should add: %AXISHOME%\bin %JAVAHOME%\bin
<code>SDKHOME</code>	Path the top-level directory of the unpacked SDK download. For example: C:\devprojects\visdk201\SDK

Importing Server-Certificates into the Java Keystore

This step is required **only** if the target servers are configured for HTTPS (the default configuration). To use HTTP (thereby obviating the need to import certificates), you must follow the procedure detailed in [“Modifying the Server Configuration to Support HTTP”](#) on page 14 before you can attempt to execute any code against the server.

These instructions assume that the JDK is in your system CLASSPATH, as detailed above.

To import certificates into a local Java keystore

- 1 Open the Windows command prompt or Linux shell command.
- 2 Create the directory for the Java certificate store. The batch files and scripts that have been provided in the VI SDK to automate various setup tasks require the keystore to be in specific directories, as shown in [Table 3-3](#). (Create the directory only: the keystore file (`vmware.store`) is created during the process of importing the certificates, in the subsequent steps below.)

Table 3-3. Java keystore location and filename

Windows	Linux
<code>C:\VMware-Certs\vmware.store</code>	<code>~/vmware-certs\vmware.store</code>

- 3 Navigate to the directory. For example, on Windows:


```
cd c:\VMware-certs
```
- 4 Use the Java keytool utility to import a certificate. The syntax is as follows:


```
keytool -import -file <certificate-filename> -alias <server-name> -keystore vmware.keystore
```

 For example:


```
C:\VMware-Certs>keytool -import -file rui.crt -alias sdkpubs01 -keystore vmware.keystore
```

 You will be prompted to create a password for the keystore you are creating:


```
Enter keystore password:
```
- 5 Create a password for the keystore by entering it at the prompt. The keystore utility displays the certificate information at the console. For example:


```
Owner: OID.1.2.840.113549.1.9.2="1183400896,564d7761726520496e632e",
      CN=sdkpubslab-01.vmware.com, EMAILADDRESS=ssl-certificates@vmware.com,
      OU=VMware ESX Server Certificate, O="VMware, Inc.", L=Palo Alto,
      ST=California, C=US Issuer:
      OID.1.2.840.113549.1.9.2="1183400896,564d7761726520496e632e",
      CN=sdkpubslab-01.vmware.com, EMAILADDRESS=ssl-certificates@vmware.com,
      OU=VMware ESX Server Certificate, O="VMware, Inc.", L=Palo Alto,
      ST=California, C=US Serial number: 0 Valid from: Mon Jul 02 11:28:17 PDT 2007
      until: Mon Aug 31 11:28:17 PDT 2026

Certificate fingerprints:
MD5: . . .61:35:C0:C4
SHA1: 4C:...78:B2
```

 At the end of the certificate information, a prompt displays a request for confirmation that the certificate should be trusted:


```
Trust this certificate? [no]:
```
- 6 Type **yes** (and press <Enter>) to respond to the prompt and import the certificate into the `vmware.store` keystore. The console displays:


```
Certificate was added to keystore
```
- 7 Repeat this process for each target server.

Generating Stubs and Compiling Classes

The VI SDK includes a client-side `vim.jar` file (created from the `vimService.wsdl` and `vim.wsdl`) using Axis 1.2.1 and JDK 1.5. Assuming you have these same versions, you can use the `vim.jar` provided, without re-generating or re-compiling.

If you are using a different version of the JDK or Axis, you must regenerate and recompile the client-side library. You can use the provided `build.bat` (or `build.sh`) script to perform all necessary tasks for you.

These instructions require all environment variables to be set as detailed in [“Setting Environment Variables”](#) on page 20.

To generate stubs and compile using the build.bat (or build.sh) script

- 1 Open a command prompt.
- 2 Navigate to the sub-directory containing the build.bat and build.sh files:
- 3 Run the build.bat (or build.sh) script by entering its name at the command prompt.

```
build
```

Shortly, you’ll see output at the console, starting with “Generating stubs from wsdl.” In a few minutes, the entire process should be complete (you’ll see the word “Done” at the command prompt, as shown in [Example 3-1](#)).

Example 3-1. Successful stub generation and compilation using build.bat script

```
Generating stubs from wsdl
Compiling stubs.
...
Done.
C:\devprojects\visdk201\SDK\samples_2_0\Axis\java>
```

You will also note that the vim.jar and vimsamples.jar files reflect the current date and time.

You can run any of the sample applications by following the instructions in the next section, [“Running the SimpleClient Sample Application to Validate Setup.”](#)

Running the SimpleClient Sample Application to Validate Setup

You can quickly test connectivity, environment settings, and successful certificate import by running one of the pre-compiled Java classes, such as SimpleClient, which connects to the server and obtains a listing of the top-level inventory entities, their properties and references. You can run Java class files using fully syntax as shown in [To run the pre-compiled SimpleClient from the command prompt](#), or use the provided run.bat (or run.sh) to simplify the process (as detailed in [To run a sample \(or other\) application using the run.bat \(or run.sh\) script](#), below).

To run the pre-compiled SimpleClient from the command prompt

- 1 Open a Windows command prompt (or shell prompt on Linux).
- 2 Navigate to the directory containing the class files for the samples:
- 3 Invoke the Java runtime, providing the full package name of the SimpleClient, along with server URN, plus your credentials. You must also pass the Java keyStore location as a property. The complete syntax is as follows:

```
java -Djavax.net.ssl.trustStore=<keystore-path> <package-hierarchy-classname> <server-url>
<username> <password>
```

For example:

```
java -Djavax.net.ssl.trustStore=c:\\VMware-Certs\vmware.keystore -Xmx1024M
com.vmware.vimsample.simpleclient.SimpleClient https://sdkpubslab-02.eng.vmware.com/sdk root
password
```

```
java -Djavax.net.ssl.trustStore=c:\\VMware-Certs\vmware.keystore -Xmx1024M <classname>
-https://<servername> -<username> -<password>
```

NOTE If error messages are raised due to system heap or other memory issues, you can give the Java VM more memory, as follows:

```
java -Xms512M -Xmx1024M com.vmware.vimsample.simpleclient.SimpleClient
http://sdkpubslab-02.eng.vmware.com/sdk username password
```

To run a sample (or other) application using the run.bat (or run.sh) script

The run.bat (or run.sh, for Linux) script provided with the VI SDK saves you from entering details about the Java trustStore property. To execute any of the Java samples using the run.bat (or run.sh) script, simply pass the name of the Java class (the full package name), as in:

```
run com.vmware.vimsample.simpleclient.SimpleClient https://yourFQDNservername/sdk <username>
<password>
```

Example 3-2. Sample output of a successful run of SimpleClient (precompiled Java sample)

```
Object Type : Folder
Reference Value : ha-folder-vm
  Property Name : name
  Property Value : vm
Object Type : HostSystem
Reference Value : ha-host
  Property Name : name
  Property Value : sdkpubslab-02.eng.vmware.com
Object Type : ResourcePool
Reference Value : ha-root-pool
  Property Name : name
  Property Value : Resources
Object Type : Folder
Reference Value : ha-folder-host
  Property Name : name
  Property Value : host
Object Type : ComputeResource
Reference Value : ha-compute-res
  Property Name : name
  Property Value : sdkpubslab-02.eng.vmware.com
Object Type : VirtualMachine
Reference Value : 16
  Property Name : name
  Property Value : Windows_2K3_VM
Object Type : VirtualMachine
Reference Value : 32
  Property Name : name
  Property Value : Fedora_Linux_VM
Object Type : VirtualMachine
Reference Value : 48
  Property Name : name
  Property Value : Ubuntu_VM_03
Object Type : Datacenter
Reference Value : ha-datacenter
  Property Name : name
  Property Value : ha-datacenter
Object Type : Folder
Reference Value : ha-folder-root
  Property Name : name
  Property Value : ha-folder-root
```

Troubleshooting Setup Issues

If you are unable to successfully run the SimpleClient, check your environment settings and all other setup tasks. [Table 3-4](#) lists some common error messages and provides steps that you can take to resolve.

Table 3-4. Java runtime error messages

Symptom	Possible solution
Exception in thread "main" java.lang.NoClassDefFoundError: com.vmware.vim/ManagedObjectReference at com.vmware.vimsample.simpleclient.SimpleClient.create ServiceRef(SimpleClient.java:32) at com.vmware.vimsample.simpleclient.SimpleClient.main(SimpleClient.java:214)	<ul style="list-style-type: none"> ■ Verify that the CLASSPATH includes all Axis libraries, directly or through use of the AXISHOME environment variable. ■ Verify that the AXISHOME variable has been set correctly.
Caught Exception : Name : org.apache.axis.AxisFault Message : (301)Moved Permanently Trace : AxisFault faultCode: {http://xml.apache.org/axis/}HTTP faultSubcode: faultString: (301)Moved Permanently faultActor: faultNode: faultDetail: {}:return code: 301 {http://xml.apache.org/axis/}HttpErrorCode:301 (301)Moved Permanently at org.apache.axis.transport.http.HTTPSender.readFromSoc ket(HTTPSender.java:744)	<ul style="list-style-type: none"> ■ Verify that the server-certificate has been imported into the local keystore, and then execute the command using HTTPS and provide the credentials (user account, password) for the server.

Setting Up for Microsoft C# Development

4

To run any of the sample applications, or to start creating, compiling, and testing your own client applications, you'll need to configure your development workstation with the necessary Web-services-client tools and perform several other setup tasks, depending on the specifics of your environment. This chapter provides information for developers using Microsoft C# programming language. It includes these topics:

- [Requirements](#)
- [Setup Instructions for C# Development](#)
- [Running the Microsoft .NET \(C#\) Version of SimpleClient](#)
- [Troubleshooting Setup Issues](#)

Requirements

The VI SDK includes C# (.cs) source files and Microsoft Visual Studio project files (solutions, or .sln) for both Microsoft Visual Studio 2003 and Microsoft Visual Studio 2005. Web services client application development for C# requires:

- Development environment for C#, such as Microsoft Visual C#, Microsoft Visual Studio 2003, or Microsoft Visual Studio 2005.
- Microsoft .NET Framework, specifically Microsoft .NET Framework 2.0 (which is included with Microsoft Visual Studio 2005), or Microsoft .NET 1.1 (Microsoft .NET 1.1 is not recommended, given its legacy status).
- Microsoft .NET Framework 2.0 Software Development Kit. Depending on the specific version of the Microsoft Visual Studio and Microsoft .NET Framework that you use, you may need to also specifically install the Microsoft .NET Framework 2.0 software development kit, which includes the command-line C# compiler (csc.exe).



WARNING VMware strongly encourages developers to **use Microsoft Visual Studio 2005 and Microsoft .NET 2.0 only** (not legacy versions, such as Visual Studio 2003 and Microsoft .NET Framework 1.1) with the VI SDK. There is a serious performance issue that occurs when using the VI SDK 2.0.1 Visual Studio 2003 and the Microsoft .NET Framework 1.1.

Furthermore, VMware provides a patch file that resolves the issue and improves performance—the patch can be applied to Microsoft Visual Studio 2005 environments only. To obtain the patchfile, download the ATTpatch201.zip file, located in the Attachments section of Knowledge Base article 87402:

<http://kb.vmware.com/kb/87402>

Instructions about using the patch are included in this section.

Setup Instructions for C# Development

Specific setup instructions will vary, depending on whether your development workstation already meets some or all of the requirements, and whether you plan to use the provided samples.

To setup a development workstation to use C#

- 1 Install the Microsoft Visual programming environment, such as Microsoft Visual Studio 2005, Microsoft Visual Studio .NET 2003, or Microsoft Visual C#.

VMware recommends using Microsoft Visual Studio 2005, which includes the required .NET Framework 2.0 and improved versions of Web-services-client tools. For more information, visit:

<http://msdn2.microsoft.com/en-us/vcsharp/default.aspx>

- 2 Obtain the Microsoft .NET Framework 2.0 or Microsoft .NET Framework 1.1. If you have been using Microsoft development tools for any length of time, it's likely you already have what's needed. If not, you can obtain Microsoft .NET Framework from Microsoft, at:

<http://msdn2.microsoft.com/en-us/vcsharp/default.aspx>

- 3 Obtain the VMware Infrastructure SDK (VI SDK) package from VMware Web site:

<http://www.vmware.com/download/sdk/>

- 4 Unpack the various components into appropriate sub-directories. Use Microsoft defaults for Microsoft Visual Studio, Microsoft .NET Framework, and Microsoft .NET Framework SDK.

- 5 Obtain the VMware SDK 2.0.1 patchfile (click the ATTpatch201.zip link in the Attachment section of Knowledge Base article 87402, “.NET Takes a Long Time to Instantiate the VimService Class”), from:

<http://kb.vmware.com/kb/87402>

NOTE You need not read the KB, unless you want additional background information about its purpose, or you want to perform the patch process manually.

The ATTpatch201.zip link opens a download dialog. Navigate to a temporary or other directory of your choice and save the zipfile. The default name of the zip for download is:

87402_fATTpatch201.zip

- 6 Unpack the contents of the zipfile with “Use folder names” selected during the unpack. The contents of the zipfile will decompress to a sub-directory named **patch201**.
- 7 Open a Windows command-prompt and navigate to the patch201 sub-directory. For example:

```
cd c:\temp\patch201
```

The patch201 sub-directory contains 8 files, including various readmes (.txt) and several batch or command files (.cmd) that perform various tasks both during the patch process and after, at runtime. In the patchset sub-directory, you will see several readmes (README_patch.txt, GeneratingStubs.txt, for example), which you can disregard unless you want to exercise discrete control over the process.)

- 8 Run the patch201.cmd at the command prompt, passing the full path to the SDK home directory, as in:

```
C:\temp\patch201>patch201 c:\devprojects\visdk201\SDK
```

The command script renames the existing version of the Build2005.cmd and README.txt to time-and-date-stamped duplicate filenames, and copies the 8 files to the SDK home directory.

Example 4-1. Successful application of patch to DotNet environment

```

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\temp\patch201>patch201 c:\devprojects\visdk201\SDK
Backed up existing Build2005.cmd to Build2005.cmd.2007-08-05@23-00-40-840
Backed up existing README.txt to README.txt.2007-08-05@23-00-40-840
ECHO is off.
Copying patch files to c:\Data\VMware\SDKs\SDK\samples_2_0\DotNet
ECHO is off.
Does C:\Data\VMware\SDKs\6aug\SDK\samples_2_0\DotNet\README.txt specify a file name
or directory name on the target (F = file, D = directory)? f
1 File(s) copied
1 File(s) copied
1 File(s) copied
1 File(s) copied
1 File(s) copied
1 File(s) copied
1 File(s) copied
Patch Done.
ECHO is off.
C:\temp\patch201>

```

You must now generate stubs and build the client-side components by running the Build2005.cmd file, which has been copied to the %SDKHOME%\DotNet directory.

To ensure that all paths to all tools (wsdl.exe, csc.exe) are set correctly, it's best to execute the command script from the Microsoft .NET 2.0 SDK command prompt, available from the .NET SDK menu:

- a Open the Microsoft .NET Framework 2.0 SDK command prompt (from the Windows Start menu, select **Programs->Microsoft .NET Framework SDK v2.0->SDK Command Prompt**. The command prompt launches.
 - b Navigate to the sub-directory containing the Build2005.cmd and other files:


```
cd %SDKHOME%\samples_2_0\DotNet
```
- 9 Run the Build2005.cmd to generate new stubs, modify the newly generated stubs (for the performance problem), and compile. Simply enter the name of the script at the command prompt:

```
build2005
```

The result of a successful execution of this script is shown in [Example 4-2, "Example of a successful build session using the patchset's replacement Build2005.cmd,"](#) on page 30. As you can see (from the output shown in [Example 4-2](#), the script does a lot more than simply generating stubs and compiling: the performance issue (discussed in Knowledge Base article 87402) is circumvented to a large degree by using a Microsoft tool to optimize the generated stubs.

- 10 Test your development workstation. See ["Running the Microsoft .NET \(C#\) Version of SimpleClient"](#) on page 30.

Example 4-2. Example of a successful build session using the patchset's replacement Build2005.cmd

```

C:\devprojects\visdk201\SDK\samples_2_0\DotNet>build2005
Visual Studio C# 2005 Express is installed
Setting Path
Checking and Creating stage
A subdirectory or file stage already exists.
Microsoft (R) Web Services Description Language Utility
[Microsoft (R) .NET Framework, Version 2.0.50727.42]
Copyright (C) Microsoft Corporation. All rights reserved.
Writing file 'stage\VimObjects.cs'.
Microsoft (R) Visual C# 2005 Compiler version 8.00.50727.42
for Microsoft (R) Windows (R) 2005 Framework version 2.0.50727
Copyright (C) Microsoft Corporation 2001–2005. All rights reserved.

Microsoft (R) Xml Serialization support utility
[Microsoft (R) .NET Framework, Version 2.0.50727.42]
Copyright (C) Microsoft Corporation. All rights reserved.
Serialization Assembly Name: VimService2005.XmlSerializers, Version=0.0.0.0, Culture=neutral,
    PublicKeyToken=null.
Generated serialization assembly for assembly
    C:\Data\Development\visdk201_GA\SDK\samples_2_0\DotNet\vimservice2005.dll -->
    '.\VimService2005.XmlSerializers.dll'
'.
Optimizing generated stubs...
Microsoft (R) Visual C# 2005 Compiler version 8.00.50727.42
for Microsoft (R) Windows (R) 2005 Framework version 2.0.50727
Copyright (C) Microsoft Corporation 2001–2005. All rights reserved.

Stub generation Done.
ECHO is off.
Building Samples in Debug mode
Done Building optimized Stubs and all Samples

C:\devprojects\visdk201\SDK\samples_2_0\DotNet>

```

Running the Microsoft .NET (C#) Version of SimpleClient

These instructions assume that you have followed all setup instructions, including extracting content from the patch and using the new Build2005.cmd contained in that patchset, detailed in [“Setup Instructions for C# Development”](#) on page 28 of this chapter. For all intents and purposes, using Visual Studio 2003 is discouraged.

To run the SimpleClient application

- 1 Navigate to the sub-directory where the compiled object code is located. From the top-level directory of the SDK download, the directory is as follows:

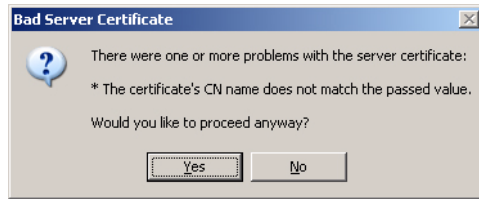
```
%SDKHOME%\samples_2_0\DotNet\cs\SimpleClient\bin\Debug
```

- 2 Run the application, passing to the command line the server URL and logon credentials. Note that, as coded, the SimpleClient application (SimpleClient.cs) requires that credentials (user account and password) be passed to the executable—even if the server has been configured to support HTTP. You can change the source code yourself and recompile.

If you don't provide all three arguments as shown here—server address, account name, and password—the executable generates an error message and comes to a halt.

```
simpleclient https://sdkpubslab-01.eng.vmware.com/sdk useraccount password
```

The application connects to the server. A “Bad Server Certificate” message displays:



- 3 Click Yes to dismiss this message and proceed to the server. Soon, the output from server should display in the console (command prompt), as shown in [Example 4-3](#):

Example 4-3. Sample output of a successful run of SimpleClient (using the

```
Object Type : Datacenter
Reference Value : ha-datacenter
  Property Name : name
  Property Value : ha-datacenter
Object Type : Folder
Reference Value : ha-folder-root
  Property Name : name
  Property Value : ha-folder-root
```

Troubleshooting Setup Issues

If you are unable to successfully run the SimpleClient, check your environment settings and all other setup tasks. [Table 4-1](#) lists some common error messages and provides steps that you can take to resolve.

Table 4-1. Microsoft C# runtime error messages

Symptom	Possible solution
SimpleClient.exe https://<management-server>/sdk <user> <pass> Caught Exception : Name : WebException Message : Unable to connect to the remote server Trace : at System.Net.HttpWebRequest.GetRequestStream() at System.Web.Services.Protocols.SoapHttpClientProtocol.In voke(String methodName, Object[] parameters) at VimApi.VimService.RetrieveServiceContent(ManagedObj ectReference _this) ... Exception disconnecting. Caught Exception : Name : NullReferenceException Message : Object reference not set to an instance of an object. Trace: ...	<ul style="list-style-type: none"> ■ Cannot connect to the web service from Microsoft .NET client sample through proxy server. Try a different server on the same subnet as the client.

Reference

This section includes these topics:

- [Batch File and Shell Script Inventory](#)
- [Java Samples](#)
- [Microsoft .NET \(C#\) Samples](#)

Batch File and Shell Script Inventory

Some of the batch files and shell scripts are not documented. The only scripts that developers might want to use are the build.bat and run.bat (on Windows) or the build.sh or run.sh (on Linux). The build.bat (and build.sh) invokes the lcp.bat (lcp.sh) and clean.bat (lcp.sh) as needed, so if you modify the batch files for any reason, be aware of the dependencies among them.

Java Development

These files included with the VI SDK download rely on environment variables for AXISHOME, JAVAHOME, SDKHOME. Note that some of the batch files are for internal use, by other batch files, and are not intended to be used by developers. Information about them is provided for completeness only.

Table A-1. Batch files and shell scripts for Java [..\Axis\java\ sub-directory]

Filename	Description	Usage note
build.bat build.sh	Checks for environment variables (AXISHOME, JAVAHOME, SDKHOME) and sets local classpath (by calling lcp.bat) using the variables as defined. The build.bat file then cleans up existing Java files (by calling clean.bat).	Use this script to generate stubs and rebuild all sample applications. Result of successful execution of build.bat is creation of vim.jar and vimsamples.jar files.
lcp.bat lcp.sh	An internal batch file that sets the local classpath on the workstation. The lcp.bat file is called by build.bat and by run.bat.	
run.bat run.sh	Batch file that enables running any of the sample applications. Sets the Java trustStore property to the local trust store and invokes the Java runtime with the name of the application passes as a parameter.	Use this script to automatically include the trustStore property setting and invoke the Java runtime with the specified class file.
clean.bat	An internal batch file is called by build.bat, to cleanup any existing artifacts prior to building the samples. Specifically, deletes all Java class files in the samples packages (vimsample.classes) and vimsamples.jar file.	

Microsoft .NET Development

The VI SDK download contains batch files for setting up the samples for Microsoft Visual Studio 2003 and Microsoft Visual Studio 2005. It also includes the project files (.sln, solution) files).

Table A-2. Batch files or command scripts for Microsoft C# .NET (..\DotNet)

Filename	Description	Usage note
Build2003.cmd	Batch file that generates client-side stubs for all sample applications for the Microsoft Visual Studio 2003 .NET development environment.	VMware recommends against using Microsoft Visual Studio 2003 (and .NET Framework 1.1).
Build2005.cmd	Batch file that generates client-side stubs for all sample applications for the Microsoft Visual Studio 2005 development environment.	VMware provides a patch (patch201.zip) that includes (among other things) a new, improved version of Build.cmd that augments the serialization code for Microsoft Visual Studio 2005 and .NET Framework 2.0 client side Web services. Applying the patch is included in the instructions in this Developer's Setup Guide. Make sure you use the improved Build2005.cmd from the patch set, rather than the original version.

Java Samples

Java samples have been compiled and tested using the tools from the Apache Axis Web-services-client libraries, version 1.2.1.

[Table A-3](#) lists the contents of the %SDKHOME%\samples_2_1\Axis\java\com\vmware\vimsample sub-directory and provides additional information about the applications contained in each folder (Java package name). The sub-directories may also include utility class or helper files.

In addition to the samples listed in [Table A-3](#), you will also find a sub-directory named “wrapper” (%SDKHOME%\samples_2_0\Axis\java\wrapper)—disregard the code in this directory: it doesn't work.

Table A-3. Java samples listing

Directory	Description	Java filename
alarms	Creates an Alarm to monitor the power state of the specified virtual machine.	VMPowerStateAlarm.java
browser	Obtains content of the specified service, starting with the root folder, and displays listing of all properties.	Browser.java
clientutils	Basic client-side utility classes that support logging, connecting to the service. Used by the other samples to handled basics (connect, login, logout).	ClientBase.java ClientInfo.java ClientUtil.java Log.java ServiceConnection.java ServiceUtil.java
connect	Connect to specified service, login using specified credentials, logout.	Connect.java
createdelete	Creates Folder, Datacenter, and Cluster managed entities.	Create.java
	Deletes Folder, Datacenter, and Cluster managed entities.	Delete.java
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-3. Java samples listing

Directory	Description	Java filename
hostops	Do not use—not functional [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
	Utility class used by all	NicUtils.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
license	Demonstrates uses of the Licensing API.	LicenseManager.java
moverename	Moves the specified Managed Entity,.	Move.java
	Renames a Managed Entity	Rename.java
objectdiscovery	Demonstrates using the PropertyCollector managed object.	PropertyCollector.java
	Demonstrates using the SearchIndex managed object.	SearchIndex.java
performance		Basics.java
	Reads performance measurements from the past	History.java
		PrintCounters.java
	Reads performance measurements from the current time	Realtime.java
		Util.java
	An ESX Top look-alike demonstrating VI Performance monitoring API. Note: This sample is non-functional at the present time.	VI Top.java
	Demonstrates using the PerformanceManager managed object type.	VIUsage.java
		widgets \ LineChart.java widgets \ StatsTable.java
remove	Removes or un-registers an Managed Object Note: Does not delete Hosts at the present time.	RemoveManagedObject.java
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
simpleclient	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
tasks	Displays details about currently running tasks.	TaskList.java
tools	Verifies PropertySpec data and writes an object model and helper functions in Java (or C#) code for calling the PropertyCollector and initializing or updating the objects.	CreatePropertyObject.java

Table A-3. Java samples listing

Directory	Description	Java filename
updates	Displays update information from various virtual machines and hosts.	GetUpdates.java
virtualmachines	Do not use—not functional. Should demonstrate cloning a virtual machine.	VMClone.java
	Locates multiple virtual machines in a Datacenter, and does virtual machine operations like Snapshot, poweroff, etc. Note: This sample is non-functional at the present time.	VMPowerOps.java
vmcreate	Create a virtual machine.	VmCreate.java
vmpowerops	There are two versions of VmPowerOps—one basic, one advanced. The advanced version does not work. The basic VmPowerOps obtains a reference to a virtual machine and invokes various power operations on the reference, as specified on the command line options.	VmPowerOps.java
vmreconfig	Reconfigure a virtual machine.	VmReconfig.java
vmsnapshot	Do not use—not functional. Should perform virtual machine snapshot operations.	VmSnapshot.java
vmutils		VmUtils.java

Microsoft .NET (C#) Samples

The C# samples include the Microsoft Visual Studio (2003) project files. These C# sample applications have been developed and tested using:

- Microsoft Visual Studio 2005, with Microsoft .NET Framework version 2.0 and the Microsoft .NET 2.0 Software Development Kit
- Microsoft Visual C# 2005 Express, with Microsoft .NET Framework version 2.0 and the Microsoft .NET 2.0 Software Development Kit

Table A-4 lists the contents of the `samples_2_0\DotNet\cs` sub-directory and provides additional information about the applications contained in each folder (namespace). The sample code sub-directories may also include Microsoft 2005 (or Microsoft Visual Studio 2003) project files (*SampleName2005.cs*, .NET assembly files (*AssemblyInfo.cs*), and ancillary class files, such as utility classes required to support the sample application.

NOTE The Microsoft C# samples do not work through a proxy server to the target server.

In addition to the samples contained in `samples_2_0\DotNet\cs`, you'll also find a sub-directory "wrapper" and a sub-directory "vb." Disregard these two sub-directories: the vb folder is empty, and the wrapper samples does not work.



CAUTION Note that various readme files in the VI SDK download may refer to Microsoft Visual Studio 2003 .NET and Microsoft .NET Framework version 1.1, but due to a performance issue with these legacy technologies, VMware does not recommend using them with the VI SDK 2.0.1 (or subsequent releases).

Table A-4. C# samples listing

Directory	Description	Source code and utility class filenames
Browser	Retrieves ServiceContent managed object and displays all properties, starting from the root folder. To obtain properties for a specific type, or by default, for ManagedEntity.	Browser.cs

Table A-4. C# samples listing

Directory	Description	Source code and utility class filenames
ClientUtils	Utility classes (that are used by other samples, as required) that provide client-side infrastructure (setup connection, log on to service, and so on).	CertPolicy.cs ClientBase.cs ClientInfo.cs ClientUtil.cs Log.cs ServiceUtil.cs SvcConnection.cs
Connect	Connects to the service and logs in, using provided credentials. Displays listing of ServiceContent managed object. Logs out and disconnects from service.	Connect.cs
EventFormat	Retrieves the most recently captured event and formats it for display. Demonstrates Event formatting.	EventFormat.cs
MobStartPage		MobStartPage.cs
SimpleClient	Basic sample that connects to specified server and logs on; retrieves content listing, and then closes the connection and logs out.	CertPolicy.cs SimpleClient.cs
VmPowerOps	Retrieves a reference to a vm and invokes various power operations on it, as specified on the command line options.	VmPowerOps.cs
WatchVM	Uses the PropertyCollector managed object to monitor updates associated with a specified virtual machine.	WatchVM.cs

Index

A

Apache Axis
 described **19**
 generating stubs **22**
Axis **19**

C

compatibility, VI SDK versions and VMware Infrastructure
 3 versions **9**

D

Development workstation, defined **3**

H

http protocol, configuring **14**

K

knowledge base
 accessing **4**

N

non-SSL, configuring server for HTTP **14**

S

SOAP, defined **6**

T

Target server, defined **3**
target servers **11**
technical support resources **4**

U

user groups
 accessing **4**

V

VMware community forums
 accessing **4**

W

Web Services Interoperability Organization (WS-I) Basic
 Profile 1.0 **5**
WSDL (web services description language), defined **6**

