
SWsoft

VZAgent

SOAP Tutorial

1.0



SWSOFT™

(c) 1999-2007

ISBN: N/A
SWsoft
13755 Sunrise Valley Drive
Suite 325
Herndon, VA 20171
USA
Tel: +1 (703) 815 5670
Fax: +1 (703) 815 5675

© 1999-2007 SWsoft. All rights reserved.

Distribution of this work or derivative of this work in any form is prohibited unless prior written permission is obtained from the copyright holder.

Virtuozzo, Plesk, HSPcomplete, and corresponding logos are trademarks of SWsoft.

Virtuozzo is a patented virtualization technology protected by U.S. patents 7,099,948; 7,076,633; 6,961,868.

Patents pending in the U.S.

Plesk and HSPcomplete are patented hosting technologies protected by U.S. patents 7,099,948; 7,076,633.

Patents pending in the U.S.

Intel, Pentium, and Celeron are registered trademarks of Intel Corporation.

IBM DB2 is a registered trademark of International Business Machines Corp.

MegaRAID is a registered trademark of American Megatrends, Inc.

PowerEdge is a trademark of Dell Computer Corporation.

Contents

Preface	4
Documentation Conventions.....	4
Typographical Conventions.....	4
Shell Prompts in Command Examples	5
General Conventions	5
Feedback.....	5
Introduction	6
C# Tutorial	7
Choosing a Development Project.....	7
Generating Code	7
The Beginning	8
Start	8
Certificates Policy Preparation	9
Logging in and Creating a Session	10
Connection.....	11
Managing Virtual Environments.....	12
Getting a List of Environments.....	13
Getting Environment Info.....	14
Starting, Stopping, Restarting.....	14
Creating an Environment.....	14
Destroying an Environment.....	14
Using Performance Monitor	15
Guidelines	15
Perl Tutorial	16
PHP Tutorial	17
Index	18

CHAPTER 1

Preface

In This Chapter

Documentation Conventions.....	4
Feedback	5

Documentation Conventions

Before you start using this guide, it is important to understand the documentation conventions used in it. For information on specialized terms used in the documentation, see the Glossary at the end of this document.

Typographical Conventions

The following kinds of formatting in the text identify special information.

Formatting convention	Type of Information	Example
Triangular Bullet(➤)	Step-by-step procedures. You can follow the instructions below to complete a specific task.	<i>To create a VE:</i>
Special Bold	Items you must select, such as menu options, command buttons, or items in a list.	Go to the Resources tab.
<i>Italics</i>	Titles of chapters, sections, and subsections.	Read the Basic Administration chapter.
	Used to emphasize the importance of a point, to introduce a term or to designate a command line placeholder, which is to be replaced with a real name or value.	These are the so-called <i>EZ templates</i> . To destroy a VE, type <code>vzctl destroy veid</code> .
Monospace	The names of commands, files, and directories.	Use <code>vzctl start</code> to start a VE.
Preformatted	On-screen computer output in your command-line sessions; source code in XML, C++, or other programming languages.	<code>Saved parameters for VE 101</code>
Monospace Bold	What you type, contrasted with on-screen computer output.	<code># rpm -V virtuo-release</code>
CAPITALS	Names of keys on the keyboard.	SHIFT, CTRL, ALT

KEY+KEY	Key combinations for which the user must press and hold down one key and then press another.	CTRL+P, ALT+F4
---------	--	----------------

Shell Prompts in Command Examples

Command line examples throughout this guide presume that you are using the Bourne-again shell (bash). Whenever a command can be run as a regular user, we will display it with a dollar sign prompt. When a command is meant to be run as root, we will display it with a hash mark prompt:

Bourne-again shell prompt	\$
Bourne-again shell root prompt	#

General Conventions

Be aware of the following conventions used in this book.

- Chapters in this guide are divided into sections, which, in turn, are subdivided into subsections. For example, **Documentation Conventions** is a section, and **General Conventions** is a subsection.
- When following steps or using examples, be sure to type double-quotes ("), left single-quotes ('), and right single-quotes (') exactly as shown.
- The key referred to as RETURN is labeled ENTER on some keyboards.

The root path usually includes the `/bin`, `/sbin`, `/usr/bin` and `/usr/sbin` directories, so the steps in this book show the commands in these directories without absolute path names. Steps that use commands in other, less common, directories show the absolute paths in the examples.

Feedback

If you spot a typo in this guide, or if you have thought of a way to make this guide better, we would love to hear from you!

If you have a suggestion for improving the documentation (or any other relevant comments), try to be as specific as possible when formulating it. If you have found an error, please include the chapter/section/subsection name and some of the surrounding text so we can find it easily.

Please submit a report by e-mail to userdocs@swsoft.com.

Introduction

CHAPTER 2

C# Tutorial

In This Chapter

Choosing a Development Project.....	7
Generating Code	7
The Beginning.....	8
Managing Virtual Environments.....	12
Using Performance Monitor.....	15
Guidelines	15

Choosing a Development Project

You can use any kind of Visual Studio .NET C# project for your application. Your choice depends on your application requirements only. For our example, let's select a C# Windows Console Application and call it CSSoap.

Generating Code

To generate classes in Visual Studio, you need to use the Add Web Reference dialog box. The common procedure for adding a web reference to a project is as follows:

- 1 In Solution Explorer, select your project.
- 2 On the "Project" menu, choose "Add Web Reference". The Add Web Reference dialog box opens.
- 3 Type or copy and paste in the "URL" field of the dialog box the following URL:
`http://www.swsoft.com/webservices/vza/4.0.0/VZA.wsdl`
- 4 Press the "Go" button. In the web services list, you will see a single entry: "1 Service Found: VZA".
- 5 Type the desired name for the web service in the "Web reference name" field. This name will be used in your code as the C# namespace to access the selected service. In this example, we are going to call it VZA.

Press the "Add Reference" button. You'll see a new item called "VZA" appear under "Web References" in Solution Explorer.

The Beginning

Start

At the beginning, you have the generated code in your Class1.cs file looking like this:

Class1.cs:

```
using System;

namespace CSSoap
{
    /// <summary>
    /// Summary description for Class1.
    /// </summary>
    class Class1
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main(string[] args)
        {
            //
            // TODO: Add code to start application here
            //
            Console.Read();
        }
    }
}
```

We added `Console.Read()` at the end of `Main()` to let our window stay open until any key is pressed.

Certificates Policy Preparation

Since VZAgent SOAP uses HTTPS as transport protocol, we have to deal with the certificate issues. For now, we're going to use Trust All Certificates Policy. We'll create a class that implements such a policy for us and pass it to the CertificatePolicy manager during login. Here is the class:

```
// Class, used as a Certificate policy provider and allowing all
certificates
public class TrustAllCertificatePolicy : System.Net.ICertificatePolicy
{
    public TrustAllCertificatePolicy()
    {}

    public bool CheckValidationResult(System.Net.ServicePoint sp,
System.Security.Cryptography.X509Certificates.X509Certificate
cert,
System.Net.WebRequest req, int problem)
    {
        // Trust everything
        return true;
    }
}
```

Logging in and Creating a Session

First of all, we'll have to log into VZAgent, let it authenticate us and create the session ID, which we'll be using for all the subsequent operations. Please keep in mind that a created session expires after some inactivity period (3 hours by default). Of course we have to know some VZAgent login credentials. The superuser login is "vzagent0". The password for this login you were asked to enter during the creation of the Service VPS on your Virtuozzo node. Let's suppose it is "mypassword" and use it here.

```
// Function provides initial login to url with user and password
private string Login(string url, string user, string password)
{
    // Allow any certificates.
    System.Net.ServicePointManager.CertificatePolicy = new
    TrustAllCertificatePolicy();

    // Create a class, providing access to TicketM Operator of
    VZAgent.
    VZA.ticketmBinding ticketm = new VZA.ticketmBinding();

    // Set up the server URL.
    ticketm.Url = url;

    // Set up login credentials, using Encoding.GetBytes to convert
    password into byte array.
    VZA.login loginCred = new VZA.login();
    System.Text.Encoding ascii = System.Text.Encoding.ASCII;
    loginCred.user = user;
    loginCred.password = ascii.GetBytes(password);

    // Use dummy packet header
    ticketm.packet_header = new VZA.packet_headerType();

    // Invoke the login command.
    VZA.loginResponse lr = ticketm.login(loginCred);

    // Return the resulting Session ID.
    return lr.ticket;
}
```

The above function allows for initial login and returns the generated session ID. We'll invoke it from Main() and print out the result. The sequence of our operations will be invoked from the Run() function:

```
private string m_sid;
private string m_url;

// Main Function for our test.
public void Run()
{
    // Log in and store the resulting Session ID.
    m_sid = Login("https://MyServiceVPS:4646/", "vzagent0",
    "mypassword");
    Console.WriteLine("Session ID: " + m_sid);
}
```

We introduced the fields m_sid and m_url to store the connection data for future operations. Let's also add exception handling to Main() and print out the exception text to the console:

```
static void Main(string[] args)
{
    //
    // TODO: Add code to start application here
    //
    Class1 VZAServer = new Class1();
    try {
        VZAServer.Run();
    }
    catch(System.Web.Services.Protocols.SoapException ex)
    {
        Console.WriteLine(ex.Code.ToString() + ", " + ex.Message);
        Console.WriteLine("Details:" + ex.Detail.InnerText);
    }
    catch(System.Xml.XmlException xmlEx)
    {
        Console.WriteLine(xmlEx.ToString());
    }
    catch(System.InvalidOperationException opEx)
    {
        Console.WriteLine(opEx.Message + "\n" + opEx.InnerException);
    }
    Console.Read();
}
```

Now you can compile and start your program. Your output should look like this:

```
Session ID: vzagent0.30000.2c41b96ceet3768e77b
```

If you see an error, please consult the Troubleshooting Section at the back of this document.

To log out, you can use `ticketmBinding::logout`.

Connection

The VZAgent server listens for SOAP requests on port 4646. This is the port with which your connection should be made. Note also that the connection should be redirected to the Service VPS of the server you're requesting a connection with. For our example, we'll use the URL `https://MyServiceVPS:4646`.

CHAPTER 3

Managing Virtual Environments

In This Chapter

Getting a List of Environments	13
Getting Environment Info	14
Starting, Stopping, Restarting	14
Creating an Environment	14
Destroying an Environment	14

Getting a List of Environments

Let's begin using VZAgent services and accessing Virtuozzo. As a starter we'll fetch the list of the VEs that exist on the server. This can be done with the `list_ve` call of the VZAgent HWM interface. In SOAP-generated classes, the `list_ve` call corresponds to `hwmBinding.list_ve` method.

We also need a helper function to initialize bindings, set up the URL and the session ID:

```
// Initialize new binding with Url and session ID.
private System.Object InitBinding(System.Type bindingType)
{
    System.Object Binding =
bindingType.GetConstructor(System.Type.EmptyTypes).Invoke(null);

    bindingType.GetProperty("Url").SetValue(Binding, m_url, null);
    VZA.packet_headerType header = new VZA.packet_headerType();
    header.session = m_sid;
    bindingType.GetField("packet_header").SetValue(Binding, header);
    return Binding;
}
```

Once we have this helper function, we can write the code for fetching:

```
// Fetch VPS list
private string FetchVPSList()
{
    // Initialize hwmBinding reflecting HWM Operator of VZAgent
    VZA.hwmBinding hwm =
(VZA.hwmBinding)InitBinding(typeof(VZA.hwmBinding));

    // Retrieving list of VPSes for an empty list of VEID, which means
to retrieve the whole list.
    VZA.list_ve lve = new VZA.list_ve();
    long[] lstve = hwm.list_ve(lve);

    // Serializing the resulting list into a string.
    String resp = "";
    for (int i = 0; i < lstve.Length; ++i)
    {
        resp += "VEID:" + lstve[i] + "\n";
    }

    return resp;
}
```

To invoke `FetchVPSList()`, we need to add it to the very end of `Run()`, our main function:

```
// Fetch and write VPS list.
Console.WriteLine("VPS list:");
Console.Write(FetchVPSList());
```

Finally, we start fetching the VPSs:

```
Session ID: vzagent0.30000.1dc41b999eat449bc702
VPS list:
VEID:1
VEID:101
VEID:102
VEID:103
```

```
VEID:104  
VEID:105  
VEID:200  
VEID:301  
VEID:444  
VEID:517  
VEID:520  
VEID:777
```

Getting Environment Info

Starting, Stopping, Restarting

Creating an Environment

Destroying an Environment

CHAPTER 4

Using Performance Monitor

Guidelines

CHAPTER 5

Perl Tutorial

CHAPTER 6

PHP Tutorial

Index

C

C# Tutorial • 7
Certificates Policy Preparation • 9
Choosing a Development Project • 7
Connection • 11
Creating an Environment • 14

D

Destroying an Environment • 14
Documentation Conventions • 4

F

Feedback • 5

G

General Conventions • 5
Generating Code • 7
Getting a List of Environments • 13
Getting Environment Info • 14
Guidelines • 15

I

Introduction • 6

L

Logging in and Creating a Session • 10

M

Managing Virtual Environments • 12

P

Perl Tutorial • 16
PHP Tutorial • 17
Preface • 4

S

Shell Prompts in Command Examples • 5
Start • 8
Starting, Stopping, Restarting • 14

T

The Beginning • 8
Typographical Conventions • 4

U

Using Performance Monitor • 15