

Installation and Configuration Guide

Oracle AutoVue 20.1, Client/Server Deployment

ORACLE

February 28, 2011

Copyright © 1999, 2011, Oracle and/or its affiliates. All rights reserved.

Portions of this software Copyright 1996-2007 Glyph & Cog, LLC.

Portions of this software Copyright Unisearch Ltd, Australia.

Portions of this software are owned by Siemens PLM © 1986-2008. All rights reserved.

This software uses ACIS® software by Spatial Technology Inc. ACIS® Copyright © 1994-1999 Spatial Technology Inc. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS

Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007).

Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

This software and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third party content, products and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third party content, products or services.

PREFACE.....	6
Audience	6
Documentation Accessibility	6
Accessibility of Code Examples in Documentation	6
Accessibility of Links to External Web Sites in Documentation	6
TTY Access to Oracle Support Services	6
Related Documents.....	6
Conventions.....	7
INTRODUCTION	8
Installation Checklist	8
AUTOVUE SYSTEM REQUIREMENTS	9
Hardware Requirements	9
System Requirements.....	10
PREREQUISITES	12
Windows Prerequisites.....	12
Linux Prerequisites	12
BASIC AUTOVUE SERVER INSTALLATION	13
Upgrading from AutoVue Version 20.0	13
Upgrading from AutoVue Version 19.3 or Earlier	13
Installing the AutoVue Server.....	14
Deploying AutoVue in Virtualized Environments	16
Verifying Your AutoVue Installation	16
Verifying AutoVue Server Startup	16
Verifying Communication with AutoVue	17
VueServlet Deployment Instructions.....	19
Creating a WAR for the VueServlet.....	19
Deploying the VueServlet	21
Verifying VueServlet Deployment.....	21
Installing AutoVue Client Components	22
Verifying AutoVue Client	22
Troubleshooting AutoVue Client	23
INSTALLING IN AN INTEGRATED ENVIRONMENT	24
Installing AutoVue Client Components in an Integrated Environment	24
Installing the VueServlet in an Integrated Environment	24
Verifying your Integration	25
CONFIGURING AUTOVUE SERVER FARM FOR HIGH USAGE.....	26
Setting Up AutoVue Server Load Balancing	26
Symbol Libraries	26
Verifying AutoVue Server Load Balancing	26
Troubleshooting AutoVue Server Load Balancing	27
Configuring VueServlet Load Balancing	27
FAILOVER AND DISASTER RECOVERY	28
AutoVue Server Configuration for Failover.....	28
AutoVue Failover Configuration on the VueServlet.....	28
Failover for the VueServlet	28
Failover for AutoVue client components.....	29
Verifying Failover Configuration	29
INTEGRATING WITH A DMS	30
Backward Compatibility Options	30

Multiple Document Repositories.....	30
Creating an Intellistamp Template.....	31
Designing an Intellistamp Template.....	31
Configuring Intellistamp Templates.....	32
Configuring Intellistamp with Your Integration.....	33
Verifying Your Integration.....	34
OFFLINE/DISCONNECTED USE OF AUTOVUE	35
AutoVue in Online/Offline Mode	35
Configuring for Offline Mode	35
Enabling Markups for Office Formats in Offline Mode.....	35
AutoVue Mobile.....	35
Installing AutoVue Mobile.....	36
Defining Markup Policy	36
CONFIGURING FOR REAL-TIME COLLABORATION.....	39
Default Collaboration Configuration	39
Distributed Geographies Configuration.....	39
Distributed DMS Configuration	39
Configuring Across Multiple Firewalls and AutoVue Servers	40
STARTING THE AUTOVUE SERVER.....	43
Starting AutoVue on Windows	43
Starting AutoVue on Linux	43
Shutting Down the AutoVue Server	44
Running the AutoVue Server as a Service	44
On Windows OSes.....	44
On Linux OSes	45
MONITORING THE AUTOVUE SERVER	46
AutoVue Server Console.....	46
Usage Monitoring	47
Logging for the AutoVue Server.....	48
Log4j Appenders	48
Logger Information.....	50
CUSTOMIZING THE AUTOVUE CLIENT	52
AutoVue Applet Parameters	52
Scripting the Applet.....	57
Configuring a Directory-Browsing Servlet for the AutoVue Client	61
Customizing the GUI.....	62
Choosing the GUI File.....	62
Modifying the GUI	62
UNC File Names	68
Customizing the Example AutoVue Client Pages	69
SECURITY CONSIDERATIONS FOR AUTOVUE	70
Installing AutoVue	70
Clustered Deployments	70
File Permissions	70
VueServlet	70
Enabling SSL Communication.....	70
SSL Between the AutoVue Client and the VueServlet	70
SSL Between the VueServlet and the AutoVue Server.....	71
Identity Management Systems	71

NTLM Authentication Protocol	72
Integrations with AutoVue	72
AUTOVUE SERVER CONFIGURATION OPTIONS	74
AutoVue Host Name Option.....	74
RMI and Socket Ports Options	74
Process Pool Size Option.....	75
Proxy Connection Options.....	75
Streaming Files Options.....	76
DMS Options.....	77
Collaboration Options.....	77
Offline Mode Option	78
NTLM Authentication Option	78
log4j and Diagnostics Options	78
Reboot Option.....	79
Recovery Attempt Option.....	79
DLL Version Option	79
File Format Information Option.....	79
Global User Options.....	80
Markup Options	80
Server Viewable Local Files Options.....	81
Online Help Options.....	81
Memory Optimization.....	82
Linux-Specific Options.....	83
Preload Java Class Option	83
Xvfb Options	83
WINE Options	84
OEM Copyright Notice	84
VUESERVLET CONFIGURATION OPTIONS	85
APPENDIX A: DEPLOYING THE VUESERVLET ON APPLICATION SERVERS	86
Deploying the VueServlet on non-J2EE Application Servers	87
Setting up the VueServlet.....	87
Deploying on Jetty.....	88
Configuring the ISAPI Redirector on Windows IIS	88
APPENDIX B: NON-INTERACTIVE INSTALLATIONS	89
Installation	89
Uninstallation.....	90
APPENDIX C: CONFIGURING AUTOVUE PLUG-IN FOR ENTERPRISE MANAGER ..	92
Prerequisites.....	92
Installing the Plug-in	92
APPENDIX D: SAMPLES AND API EXAMPLES INCLUDED WITH AUTOVUE	94
API Examples	94
Sample Files	94
FEEDBACK	96
General Inquiries.....	96
Sales Inquiries.....	96
Customer Support	96

Preface

The *Oracle AutoVue Installation and Configuration Guide* describes how to install and configure Oracle AutoVue and its associated components.

For the most up-to-date version of this document, go to the AutoVue Documentation Web site on the Oracle Technology Network (OTN) at <http://www.oracle.com/technetwork/documentation/autovue-091442.html>.

Audience

The *Oracle AutoVue Installation and Configuration Guide* is directed at any user whose task is the installation and administration of Oracle AutoVue.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format on the Oracle Technology Network (OTN), and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at <http://www.oracle.com/accessibility/>.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, 7 days a week. For TTY support, call 800.446.2398. Outside the United States, call +1.407.458.2479.

Related Documents

For more information, see the following documents in the Oracle AutoVue documentation library:

- *Planning Guide*
- *Viewing Configuration Guide*
- *AutoVue Testing Guide*
- *User's Manual*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in the text.
<i>italic</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
monospace	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.
[root directory]\[sub directory]	In Windows and Linux OSes, directory hierarchy is written with backward slashes (\) and forward slashes (/), respectively. In this document, unless mentioned otherwise, directory hierarchy for Windows and Linux OSes are written with the backward slash.
<angular brackets>	Indicates required entries but are not to be included in the entered information.
{curly braces}	Indicates mandatory information.
[square brackets]	Indicates optional syntactical elements.
	Indicates an either-or type of choice.
...	Indicates that information may be repeated.

Introduction

AutoVue is Oracle's suite of Enterprise Visualization solutions, which are designed to view, digitally annotate and collaborate on any digital information in an organization. AutoVue delivers visualization capabilities for hundreds of document types, including business documents such as Office and Graphics, as well as technical document types such as 2-D/3-D Computer Aided Design (CAD) and Electronic Design Automation (EDA).

The Client/Server Deployment of AutoVue has AutoVue installed on a server, to which client machines connect to access and view documents. The Client/Server deployment provides a complete, open and standards-based set of integration tools that allows customers to tie AutoVue to any enterprise applications to provide users with a consistent view of data and business objects and expand workflow automation to document-based processes.

This document provides instructions for installing and configuring AutoVue Client/Server deployment. Refer to the *Planning Guide* for information on how to plan your AutoVue deployments.

Installation Checklist

The Oracle AutoVue Client/Server Deployment is a multi-tiered client-server architecture. An AutoVue solution has several components: the AutoVue server, an application server hosting the VueServlet, a Web server or an application server hosting AutoVue client components, and the AutoVue client.

AutoVue can be deployed in a number of scenarios. An AutoVue installation consists of installing the AutoVue server, VueServlet, and AutoVue client components. Refer to the following sections for more information:

["Basic AutoVue Server Installation"](#)

["VueServlet Deployment Instructions"](#)

["Installing AutoVue Client Components"](#)

Depending on your deployment scenario, additional installations and configurations may be required. The following table lists possible AutoVue deployment scenarios and their suggested installations/configurations.

Note: For more detailed information on AutoVue deployment options, refer to the *Planning Guide*.

Deployment Scenario	Jump to Chapter
Scaling AutoVue for high volume of concurrent users.	"Configuring AutoVue Server Farm for High Usage"
Planning for failover and disaster recovery.	"Failover and Disaster Recovery"
Integrating with a Document Management System (DMS) ^a .	"Integrating With a DMS"
Achieving secure communication.	"Security Considerations for AutoVue"
Secure Sockets Layer (SSL)	"Security Considerations for AutoVue"
Real-Time Collaboration across Firewalls	"Configuring for Real-Time Collaboration"
Customizing GUI/AutoVue Client	"Customizing the AutoVue Client"
Usage Logging/Server Logging	"Monitoring the AutoVue Server"
Working in a Disconnected Environment	"Offline/Disconnected Use of AutoVue"
AutoVue Plug-in for Oracle Enterprise Manager	"Appendix C: Configuring AutoVue Plug-in for Enterprise Manager"

a. The terms *document repository* and *DMS* are used interchangeably in this document when referring to DMS/ERP/PLM/UCM systems.

AutoVue System Requirements

Hardware Requirements

Component	Oracle-Certified Hardware Requirements
Server	<p>Note: The AutoVue server is very CPU-, I/O-, memory-, and graphics-intensive. Ensure that the machine running the AutoVue server is not being used for other applications. For optimal performance, we recommend that the machine running the AutoVue server should not be used by other applications.</p> <ul style="list-style-type: none"> • 8 GB of RAM • Quad-core processor • 400 MB of disk space for installation • At least 30 GB of free disk space: <ul style="list-style-type: none"> • 20 GB for streaming files (if you configure a larger size for AutoVue cache directory, ensure that the additional disk space is available). • Additional space required for managing markup symbols, user profiles, and markups. • AutoVue also stores temporary files (at the %TEMP% path on Windows operating systems and at \$TMPDIR path on Linux operating systems). These files are generally deleted after processing is complete. Ensure that there is available disk space for AutoVue temporary files.
Client	<ul style="list-style-type: none"> • 1GHz CPU • 1 GB of RAM <p>It is recommended that the Java Virtual Machine (JVM) used for the AutoVue client is configured for a maximum memory of 256 MB. If loading larger documents, you may need to increase this memory to a higher value (for example, 512 MB).</p> <ul style="list-style-type: none"> • The AutoVue client is a Java applet and as such works on most operating systems and browsers that support Java applets. To see what is certified by Oracle, refer to "System Requirements". • When running the AutoVue client on machines with non-Windows operating systems (OSes), ensure that these machines have a graphics card that supports OpenGL. This is necessary for loading 3D models. • On Windows machines, it is recommended to have a graphics card with OpenGL support. In the absence of a graphics card, Windows uses its OpenGL capability which is slower as compared to having a graphics card that supports OpenGL.

System Requirements

Component	Oracle-Certified Operating Systems and Software
<p>Server</p> <p>The installation requires about 400MB of free space. Additional space will be required by AutoVue for storing other data such as streaming files and markups.</p>	<p>Windows</p> <ul style="list-style-type: none"> Windows Server 2003 32-bit Windows Server 2003 64-bit (AutoVue running in 32-bit mode) Windows Server 2008 32-bit Windows Server 2008 64-bit (AutoVue running in 32-bit mode) Windows Server 2008 R2 64-bit (AutoVue running in 32-bit mode) <p>Important: Windows 2008R2 has much improved memory handling abilities compared to Windows 2008 and Windows 2003. It is recommended that you run AutoVue on Windows 2008R2 for better memory handling and long-term stability.</p> <p>Linux</p> <ul style="list-style-type: none"> Oracle Enterprise Linux 5.4 (x86)—32-bit Oracle Enterprise Linux 5.4 (x86)—64-bit (AutoVue running in 32-bit mode) Red Hat Enterprise Linux 5.4 (x86)—32-bit Red Hat Enterprise Linux 5.4 (x86)—64-bit (AutoVue running in 32-bit mode) <p>Virtualization</p> <ul style="list-style-type: none"> VMWare Server version 2.0.2
<p>Client</p> <p>Clients running the following 32-bit Java Virtual Machines:</p> <ul style="list-style-type: none"> J2SE 6.0 update 21 J2SE 5.0 update 22 	<p>Windows OSes (XP, Vista, and 7)—32-bit and 64-bit</p> <ul style="list-style-type: none"> Internet Explorer 7—32-bit only Internet Explorer 8—32-bit only Firefox 3.5—32-bit only <p>MAC OS X 10.6</p> <ul style="list-style-type: none"> Safari 5.0 Firefox 3.5 <p>RedHat Enterprise Linux 5</p> <ul style="list-style-type: none"> Firefox 3.5 <p>Ubuntu 10.04 LTS</p> <ul style="list-style-type: none"> Firefox 3.5 <p>Solaris 10 (Sparc)</p> <ul style="list-style-type: none"> Firefox 3.5
<p>Application Server</p> <p>The VueServlet has been certified on the following application servers:</p>	<ul style="list-style-type: none"> WebLogic 9.x and up Oracle Application Server 10g Tomcat 6.x and up WebSphere 6.1 and up Jetty 6.0 and up

Component	Oracle-Certified Operating Systems and Software
Web Server The AutoVue client Web page is certified on the following Web servers:	<ul style="list-style-type: none">• Oracle HTTP Server• Windows IIS• UNIX, Apache v2 <p>The AutoVue installer detects whether a Web server is installed on the AutoVue server machine. If one of the certified Web servers is found, the required AutoVue client components are installed. If you want to install AutoVue client components manually, or are using an integration with AutoVue, refer to the "Installing AutoVue Client Components" section for information on installing the client components.</p>

Prerequisites

Prior to installing AutoVue, there are certain prerequisites that must be met. The following prerequisites are common to Windows and Linux OSes.

- The machine that is hosting the AutoVue server must have a color depth of at least 16-bits. If the machine has a lower color depth, you may run into discrepancies in color or filling when viewing, printing or converting from AutoVue.
- If you are using a load balancer, ensure that the load balancer is configured to enable session stickiness (also referred to as session persistence). Session stickiness is normally achieved through the use of browser cookies.
- Ensure that AutoVue has permission to write to the operating system's temporary directory. On Windows, this is defined by the %TEMP% environment variable and on Linux OS, it is defined by the environment variable \$TMPDIR.
- If you are installing AutoVue client components with a Web server, ensure that the installer has write permissions to the Web server docroot.

The following sections describe Windows-specific and Linux-specific prerequisites.

Windows Prerequisites

- AutoVue installs the AutoVue Document Converter print driver on Windows operating systems. Ensure that the print spooler service is enabled and that you have the permissions to install print drivers on the AutoVue server machine.

Linux Prerequisites

Note: To correctly install AutoVue on a Linux OS, it is recommended that you have basic knowledge of Linux and its administration.

- 1 Run the standard update agent on your Linux distribution (up2date) to download the latest Xvfb and Mesa files.

Note: The AutoVue server installer does not detect whether Xvfb or Mesa are installed.

- 2 Install Xvfb version 6.8.2 or later.

Make sure you install the Xvfb with XRender and GLX extensions.

- 3 Install the latest Mesa package (recommended version is 6.5.1 or later).

Note: In the event you want to use an earlier version of Mesa, it is acceptable to use the version that is included in the repository of the supported Linux distribution.

Basic AutoVue Server Installation

This chapter describes how to install AutoVue on Windows and Linux OSes.

Note: If you want to install AutoVue in non-interactive mode, refer to ["Appendix B: Non-Interactive Installations"](#).

Upgrading from AutoVue Version 20.0

If you are upgrading from AutoVue 20.0, you do not need to run the uninstaller before you install AutoVue 20.1. You can just run the installer for AutoVue 20.1. The installer detects if AutoVue 20.0 is installed on your machine. If it is installed, the installer backs up required data, uninstalls version 20.0 and then installs 20.1 to the same location.

Below is the list of data that is migrated:

- Settings in `javueserver.properties` are migrated to the new version.
- INI options from `VueServer.ini` have been migrated to `javueserver.properties`. As part of the upgrade process, the installer migrates settings from version 20.0's `VueServer.ini` to the new version's `javueserver.properties`.
- Any changes made to `default.ini` and `allusers.ini` are migrated to the new version.
- AutoVue user profiles are left as is, since these could be used with AutoVue 20.1 as well.
- Server-managed markups are left as they are since these will be read by the new version of AutoVue.
- Any stamps and stamp libraries are left as they are since they will be read by the new version of AutoVue.
- Intellistamps and definitions are left as they are since they will be read by the new version of AutoVue.
- On Linux, changes made to `<AutoVue Install Root>/config/jvuwew_config` are left as is since this will be read by the new version of AutoVue.
- Custom log settings: If you had custom log settings, these are migrated to the new version of AutoVue.
- Changes made to markup policy file are migrated to the new version of AutoVue.
- Any changes made to format-specific files such as color maps, font maps, fonts are migrated to the new version.

The following are not migrated and must be migrated manually:

- GUI files: If you created custom GUI files, they are left as is. However, you must make sure to manually migrate the GUI settings. In order to migrate GUI, it is recommended that you run a diff utility between version 20.0's `default.gui` and your custom GUI. Identify what GUI components have been updated. Manually apply these settings to the 20.1 GUI file.

Upgrading from AutoVue Version 19.3 or Earlier

If you are upgrading AutoVue from version 19.3 or earlier, you must manually move your configuration settings from your version to AutoVue 20.1. You must first uninstall any service pack(s) that are installed for that version of AutoVue and then uninstall your previous version before installing the new version of AutoVue. Before you uninstall, you will need to backup all required data. Once you install 20.1, you must migrate your past data to 20.1. Below is what you need to backup and migrate manually:

- Custom settings in `javueserver.properties`
- Custom settings in `VueServer.ini`. In version 20.1, settings in `VueServer.ini` have been migrated to `javueserver.properties`. Refer to the *Release Notes* for a mapping of `VueServer.ini` option to `javueserver.properties` parameter.
- Custom settings in `default.ini` and `allusers.ini` should be backed up and migrated to the new version.
- User-specific INI files should be backed up and copied over to the Profiles folder of the new installation.
- GUI files: If you created custom GUI files, you must make sure to migrate the GUI settings. In order to migrate GUI, it is recommended that you run a diff utility between your current version's `default.gui` and your custom GUI. Identify what GUI components have been updated. Manually apply these settings to 20.1 GUI file.
- Custom log settings: If you had custom log settings, apply them manually on the new version of AutoVue.
- Intellistamp attributes and settings from the `dmstamps.ini` file (located in the `<AutoVue Install Root>\bin` directory) should be copied over manually to the new version

- Markup files, if markups are being managed by the AutoVue server (located in the <AutoVue Install Root>\bin\Markups directory) should be backed up and copied over to the new version of AutoVue.
- Custom markup symbol libraries (located in the <AutoVue Install Root>\bin\Symbols directory) should be backed up and copied over to the new version of AutoVue.
- If MarkupPolicy.xml located at <AutoVue Install Root>\bin was modified, it should be backed up and changes to the policy should be manually applied to the new version.
- On Linux installations of AutoVue, backup the jvview_config file (located in the <AutoVue Install Root>/config directory) if it was modified and apply the changes manually to the new version.
- Any changes made to format-specific files such as color maps, font maps, fonts must be backed up and these changes should manually be applied to the new version.

Installing the AutoVue Server

Important:

- Shutdown all applications (including AutoVue) before you run the installer for the AutoVue server.
- If the installer prompts you to reboot the machine before or after the uninstallation, you must reboot the machine in order to get a successful installation.

For Linux OSes, you must install the WINE RPM package before installing the AutoVue server. To do so, install wine-av-20040914-21.i386.rpm from <http://oss.oracle.com/AutoVue>.

- If you have an older version of WINE, you need to uninstall it and then install the package that is certified with this release of Oracle AutoVue.
- Install WINE as a root user by running the following:

```
#rpm -i wine-av-20040914-21.i386.rpm
```

Note: This version of WINE is installed in the /usr/av directory.

To install the AutoVue server, do the following:

- 1 Download the Oracle AutoVue Media Pack and extract its contents.
- 2 Run the AutoVue installer executable:

Windows OS: The installer is InstallClientServer.exe.

Linux OS: The installer is InstallClientServer_lin.bin.


Note: You might need to grant execute permissions to the installer binary on Linux. To do so, run `chmod +x InstallClientServer_lin.bin`.


- 3 Select a language from the installation dialog and then click **OK**.
- 4 Click **Next** to begin installation.
- 5 Specify the installation directory and then click **Next**.

Windows OS Example: C:\Oracle\AutoVue

Linux OS Example: /home/apps/autovue

- 6 Click an installation set icon and then click **Next**:

Installation Set	Description
 Standard	Installs the most common AutoVue features. Note that this set does not install the sample drawing files or API examples.

Installation Set	Description
 Custom	You can select the features to install. Select this installation set to install the sample drawing files and API examples.

If you selected the **Custom** install set continue to step 7, otherwise go to step 8.

- 7 Select which of the following features to install and then click **Next**:

Option	Description
Program Files	Installs Oracle AutoVue. The option is selected by default.
Administration Documentation	Installs Oracle AutoVue system administration documentation. The option is selected by default.
User Documentation	Installs AutoVue end-user documentation. The option is selected by default.
Example Client Application	Installs a sample AutoVue client application. The option is selected by default.
Sample Files	Installs drawing sample files.
API Examples	Installs examples of how Oracle AutoVue features can be added to third-party applications using APIs.

- 8 For Windows OS installations, select one of the following locations to create shortcuts and then click **Next**.

Options	Description
In a new Program Group	Creates a shortcut in the Program group of the Start menu. For example, Oracle AutoVue. This is the default option.
In an existing Program Group	Adds a shortcut to an existing Program group. For example, Accessories.
In the Start Menu	Adds a shortcut in the Start menu.
On the Desktop	Adds a shortcut on the Desktop.
In the Quick Launch Bar	Adds a shortcut to the Quick Launch bar.
Other	Adds a shortcut to the specified location.
Don't create icons	Shortcuts are not created.

To create icons for all users of AutoVue, select **Create Icons for All Users**.

- 9 Specify a host name or IP address for the AutoVue server and then click **Next**.

Note: The hostname cannot include an underscore (_) character.

Example: hostname1.domain.com

- 10 Specify the hostname and port of the Web server and then click **Next**.

Example: hostname1.domain.com:80

- 11 Specify the document root of the Web server. If IIS, Oracle HTTP Server or Apache Web Server is installed, the installer detects the installation and auto-populates the document root. If you are using a Web server other than IIS/Oracle HTTP Server/Apache Web Server on the same machine as the AutoVue server, you can select **Custom** and then enter the document root of this Web server in the text area. Click **Next** when done.

Note: If the Web server is not installed on the same machine as the AutoVue server, refer to ["Installing AutoVue Client Components"](#) for more information.

Web Server Options	Description
Oracle HTTP Server	Document root example: C:\product\10.1.3.2\companionCDHome_1\ohs\htdocs
Apache Web Server	Document root example: /var/apache/htdocs
Microsoft IIS	Document root example: C:\inetpub\wwwroot
Custom	If you are using another Web server, you can specify the document root for this Web server.

- 12 Specify the path to the sample HTML and client JAR files relative to the document root of the Web server.

Default value: jVue

Note: Sample HTML pages and client Jar files are installed in this directory in the root of the Web server's tree. For example, <http://hostname1/jVue/>...

- 13 Review the pre-installation summary and then click **Install**.

The AutoVue server is installed in the specified directory. If there are any warnings or errors, refer to the installation log file, `Oracle_AutoVue_InstallLog.log`, located in the <AutoVue Install Root> directory.

Note: For information on registering and running AutoVue as a service, refer to ["Running the AutoVue Server as a Service"](#).

Deploying AutoVue in Virtualized Environments

To set up additional AutoVue servers on virtualized environments, copy an image of the virtual machine where AutoVue server is installed. Once you modify the name of the machine, change the name of the AutoVue server host in `javueserver.properties`. Refer to section ["AutoVue Host Name Option"](#) for information on changing the host name.

Verifying Your AutoVue Installation

Verifying AutoVue Server Startup

Start the AutoVue server:

- On Windows Operating systems, run *Start AutoVue Server* from the *Oracle AutoVue* programs shortcut.
- On Linux Operating systems, go to <AutoVue Install Root>/bin and run `./javueserver`.

The AutoVue server console should startup and the P, 1, 2, 3, 4 and M buttons should turn green.

If any of the P, 1, 2, 3, 4 and M buttons stay red or yellow, refer to the troubleshooting steps in section ["Troubleshooting AutoVue Server Startup Issues"](#).

Troubleshooting AutoVue Server Startup Issues

If the AutoVue console does not startup or if any of the buttons on the server console do not turn green, review the following trouble-shooting pointers. Error messages are written to the `log4j-roll*.log` file at <AutoVue Install Root>\bin\logs. Refer to the logs to determine the specific cause for the issue.

- On Linux operating systems:
 - Ensure that your Linux terminal is properly configured for graphics display. When the terminal does not support graphics, AutoVue server will startup, but the console will not appear. The message you would see on

the console in this case will be something like “No display defined; console will not be started”.

To start up the console separately, you can run `./jvueserver_debug -u`.

- If you see a fatal error message for Xvfb in the log, it indicates that there are issues starting up Xvfb. Possible reasons are that Xvfb is not installed correctly or the user account running AutoVue does not have permissions to start Xvfb or there is a port conflict for the Xvfb port.
To resolve this issue, ensure that Xvfb is correctly installed and that the user account running AutoVue has permissions to start Xvfb. If there is a port conflict, try modifying the port by modifying the port in the `xvfb.display` parameter in `jvueserver.properties`.
- If you have an incorrect version of WINE or if you do not have WINE installed, AutoVue server will not start up. The following message appears in the log file: “./jvueserver: could not locate WINE server should be at /usr/av/bin/wineserver...”. Uninstall any previous version of WINE and install the version that is compatible with the version of AutoVue server.
- If any of the pre-requisite libraries are missing from your installation, AutoVue server will not startup. If you see a message “Failed to initialize preloader class, aborting...” in the log, you must ensure that all pre-requisite libraries are installed on the machine.
- If the user account running AutoVue server does not have permissions to write into the AutoVue installation directory, AutoVue server will not start up. You will see error messages that say “Permission denied” in the AutoVue server logs. Ensure that the user account running AutoVue has write permissions into the AutoVue installation directory.

Note: On Linux, if you accidentally run AutoVue as a super-user, AutoVue will create/update files and set permissions as this super-user. If you later run AutoVue as a normal user, AutoVue server will not startup since the user does not have permissions to the files created by the super-user.

- If the host name of the AutoVue server changed since you installed AutoVue, the AutoVue server will not be able to start up. The message you will see in the log in this scenario is “Connection refused to host”. Update `jvueserver.properties` and set the correct server name in the parameter `jvueserver.hostname`. Similarly, if you installed AutoVue server using the IP address and the IP address changed after you installed AutoVue, you must set the correct IP address in parameter `jvueserver.hostname` in `jvueserver.properties`.
- AutoVue server needs RMI ports in order to run correctly. If the RMI ports required by AutoVue are used by other applications, the server will not startup. In this instance, you will see “java.rmi.connection” exceptions in the log. AutoVue needs the RMI port specified in property `jvueserver.rmi.port` and n consecutive ports following this port, where n is the AutoVue process pool size.
If the RMI ports required by AutoVue are not available, change the parameter `jvueserver.rmi.port` to point to a port that is available. Ensure that this port and n consecutive parts following this port are available to AutoVue.

Refer to section [“AutoVue Server Configuration Options”](#) for a list of all AutoVue server configuration options in `jvueserver.properties`.

If you verified all the above and AutoVue server still does not startup, contact Oracle customer support for help with trouble-shooting your AutoVue server startup issues.

Verifying Communication with AutoVue

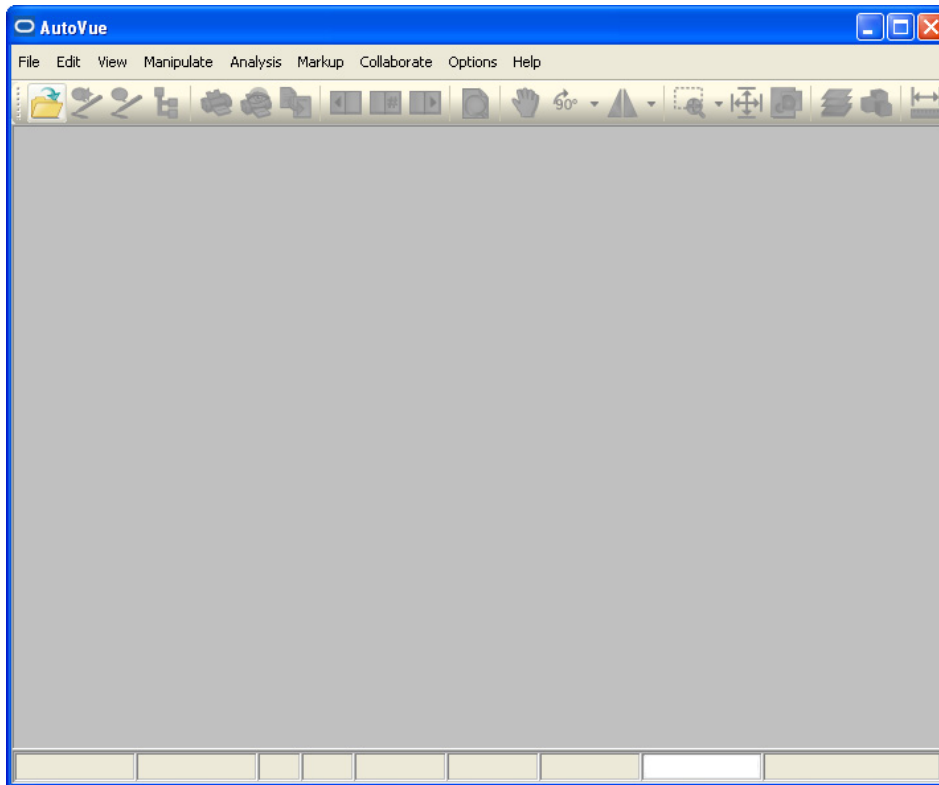
Once your AutoVue server has started up correctly, you must verify if clients can connect to the AutoVue server and you can load files in AutoVue.

- 1 For verification purposes, startup Jetty that ships with AutoVue:
 - **Start VueServlet on Jetty** from the AutoVue programs shortcut on Windows OS
 - `<AutoVue Install Root>/bin/Jetty/bin/startJetty` on Linux OS

Note: Once you finish the verification, you can shut down Jetty if you do not plan to use Jetty as part of your deployment.

- 2 Launch the example AutoVue client application by running:
 - `<AutoVue Install Root>/bin/jvue.bat` on Windows OS
 - `<AutoVue Install Root>/bin/jvue` on Linux OS

The AutoVue applet should load successfully.



- 3 Once the AutoVue client is loaded, verify if you can load files. Load a few files belonging to various format groups that you intend to load using AutoVue.

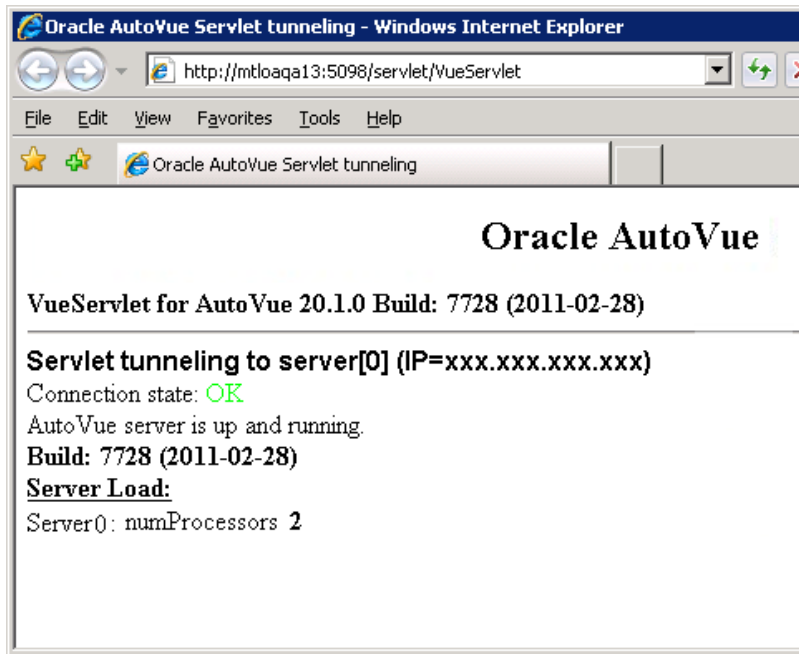
If you have issues loading the client or loading files in AutoVue, refer to the following section for trouble-shooting pointers.

Troubleshooting Communication Issues

If the AutoVue server has started up fine, but you are unable to open the client or load files, you can review the following pointers to identify other potential issues:

- If the AutoVue client does not startup, verify that the VueServlet is working properly with Jetty. Open the URL to the VueServlet in a web browser: `http://<AutoVue server host name>:5098/servlet/VueServlet`. As displayed

in the following Oracle AutoVue Servlet tunneling page, *Connection State: OK* states that the VueServlet is configured correctly.



- If the above verification fails, ensure that Jetty is working properly. If another application is using the same port as Jetty, change the port for Jetty by updating the `jetty.port` parameter in `jetty.xml`.
- Verify that the example client application is pointing to the correct URL for the VueServlet. If you modified the port for Jetty, ensure that the example client application is pointing to the right port.
- Verify that the socket port required by AutoVue is not in use by other applications. If the socket port is not available for AutoVue, modify the socket port by updating the `javueserver.socket.port` parameter in `javueserver.properties`.
If you modify the socket port, ensure that the VueServlet points to the correct socket port. Update the `JVUESERVER` parameter in `webdefault.xml` to point to the correct AutoVue server name and socket port.

VueServlet Deployment Instructions

The VueServlet is the main entry point for communications between the AutoVue clients and the AutoVue server. The client makes requests using the HTTP/HTTPS protocol to the VueServlet and the VueServlet communicates with AutoVue using AutoVue's socket ports. The instructions for deploying VueServlet vary based on whether or not you are integrating AutoVue with a DMS. This section discusses installing a single-instance of VueServlet in a non-integrated environment. In an integrated environment, the same instructions apply, except with the difference that the VueServlet may be deployed in a different context. For information on deploying VueServlet in an integrated environment, refer to section ["Installing the VueServlet in an Integrated Environment"](#).

The first step to deploying the VueServlet is to create a WAR file for the VueServlet. Once the WAR file has been successfully created, you can deploy the WAR file with your J2EE-enabled application server.

Refer to section ["Appendix A: Deploying the VueServlet on Application Servers"](#) for instructions for deploying the VueServlet with WebLogic, Tomcat, WebSphere and Jetty.

Creating a WAR for the VueServlet

To deploy the VueServlet with your J2EE-enabled application server, you must first create a WAR file. The following steps explain how to do this:

- 1 Create a directory.

For Example: C:\csiwar

- 2 In the folder C:\csiwar, create a sub-directory WEB-INF.
- 3 In WEB-INF, create a directory lib: C:\csiwar\WEB-INF\lib
- 4 Copy vueservlet.jar from <AutoVue Install Root>\bin to C:\csiwar\WEB-INF\lib.
- 5 Create a deployment descriptor file named web.xml in the WEB-INF directory.
 - The following is the mandatory header for the web.xml document. It defines the document as an XML file and relates the file syntax to the DOCTYPE resource specified.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web
Application 2.2//EN" "http://java.sun.com/j2ee/dtds/web-app_2_2.dtd">
```

- Use the following code to specify the deployment descriptor needed to deploy the VueServlet.

```
<web-app>
  <servlet>
    <servlet-name>com.cimmetry.servlet.VueServlet</servlet-name>
    <servlet-class>com.cimmetry.servlet.VueServlet</servlet-class>
    <init-param>
      <param-name>JVueServer</param-name>
      <param-value>hostname:socketport</param-value>
    </init-param>
    <init-param>
      <param-name>Verbose</param-name>
      <param-value>0</param-value>
    </init-param>
  </servlet>
  <servlet-mapping>
    <servlet-name>com.cimmetry.servlet.VueServlet</servlet-name>
    <url-pattern>/servlet/VueServlet</url-pattern>
  </servlet-mapping>
</web-app>
```

The <servlet-name> parameter is how the servlet is known within the XML file.

The <servlet-class> parameter is the fully qualified Java programming language class name of the Servlet.

The <url-pattern> parameter is how the servlet is referenced from a Universal Resource Indicator (URI).

Note: The parameter structure must follow the order in the DTD definition. For example, all <servlet> tags must be defined before any <servlet-mapping>s can be specified.

- 6 Update *hostname* in web.xml with the name of the AutoVue server machine.
- 7 Update *socketport* in web.xml with the socket port for the AutoVue server.
- 8 Specify additional configuration parameters for the VueServlet at this point. Refer to section ["VueServlet Configuration Options"](#).
- 9 To create the WAR file, use the jar utility from the Java Development Kit distribution. If you are in the root directory you created for the WAR contents (C:\csiwar), use the following command:
jar cvf VueServlet.war WEB-INF

Deploying the VueServlet

Deploy VueServlet.war into your J2EE compliant application server. Refer to the instructions available with your J2EE application server for WAR deployment information.

Verifying VueServlet Deployment

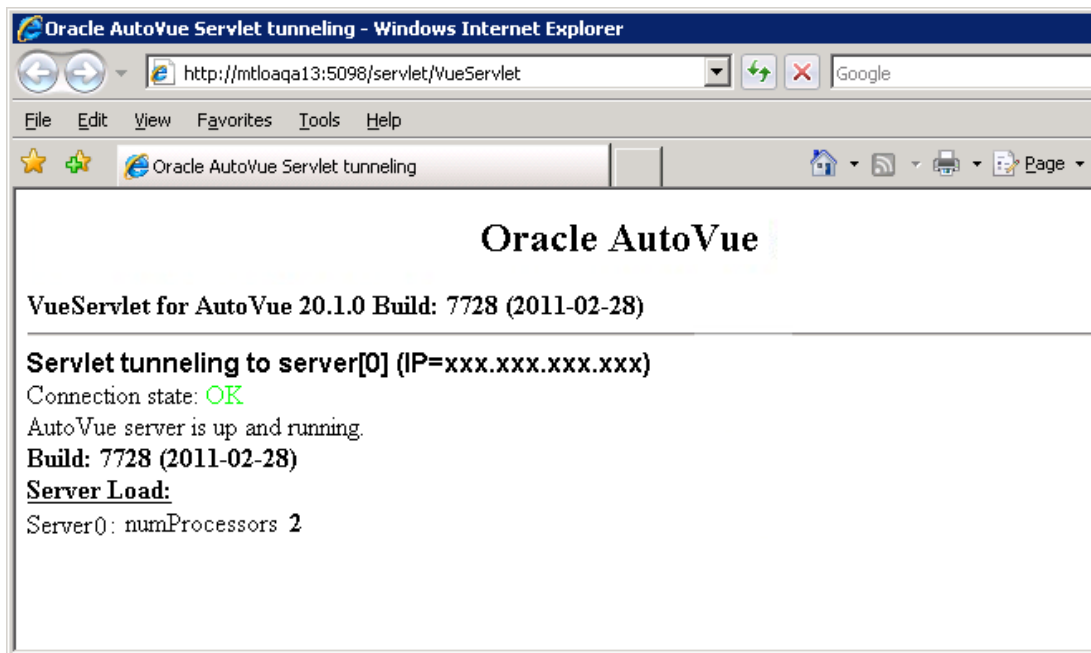
Once the VueServlet has been deployed into your application server, test the VueServlet by accessing the URL to the VueServlet. Enter the following in your web browser:

```
http://host:port/<context>/servlet/VueServlet
```

where <context> is the context you specified when deploying the VueServlet into your application server.

Note: Some application servers allow you to specify the context name, but generally the WAR file name is used as the context.

On successful deployment, the VueServlet should display a Web page as follows:



Troubleshooting VueServlet Deployment

If you are not successful with the VueServlet verification, below are some pointers to help you troubleshoot your VueServlet deployment:

- If you get a page not found error, ensure that the application server hosting the VueServlet is up and running. Check for correct syntax and verify that you are accessing the correct port for the Application server.
- If you are able to access the VueServlet page and it indicates an error connecting to the AutoVue server:
 - Verify that the AutoVue server is running
 - Verify that you specified the correct connection parameters to the AutoVue server in the VueServlet's JVUESERVER parameter
 - Verify that you can ping the AutoVue server machine from the VueServlet machine and vice-versa

Installing AutoVue Client Components

The AutoVue client is a JAVA-based applet that is the main entry point to AutoVue's capabilities. The AutoVue client components need to be made accessible to end-users at either an application server or a Web server location. The instructions for deploying the client components vary depending on whether you have AutoVue integrated with a DMS or if you are using a non-integrated environment. For information on deploying AutoVue in an integrated environment, refer to section ["Integrating With a DMS"](#).

This section discusses deploying AutoVue client components in a non-integrated environment. In an integrated environment, the same instructions apply, except with the difference that the client components may be deployed in a different location.

- 1 Create a folder (for example, named AutoVue) on your Web server docroot.
- 2 Copy all mandatory JAR files from the <AutoVue Install Root>\html directory to the directory you created on your Web server docroot. The files to copy are jvue.jar, jogl.jar, gluegen-rt.jar.
- 3 To use the sample HTML pages provided with AutoVue, copy them from the <AutoVue Install Root>\html directory to the directory you created on your Web server docroot.
- 4 Edit the files that embed the AutoVue applet and replace the following parameters with appropriate values.

Parameter	Description
CODEBASE	Specify the URL to the AutoVue client files on your Web Server (the folder created above). For example: <code>http://AutoVueClient:8080/jVue</code> Note:
JVUESERVER	Specify the servlet connection to the AutoVue server. Separate multiple values with a semi-colon. For example: <code>http://AutoVueServer:5098/servlet/VueServlet</code>

- 5 To access the sample files that ship with AutoVue, edit `frmFiles.html` and replace the values for the variable identified in table below with the appropriate value:

Variable	Description
JVUESAMPLES	Note: This feature is optional. During AutoVue installation, you must select the Sample Files check box to include the sample files. Specify the URL to the AutoVue sample files on your Web Server. For example: <code>http://AutoVueClient:8080/jVue/samples</code>

- 6 Copy the AutoVue Online help files to the docroot location. Verify the URL of the online help to make sure you are able to load the online help files.
- 7 Update the URL to the help files in AutoVue server's `javueserver.properties` file.

Verifying AutoVue Client

Once you have the AutoVue client components installed, you can verify if the installation is successful by opening the URL to the AutoVue client files. For example: `http://AutoVueClient:8080/jVue/jVue.html`.

The AutoVue client should load and the AutoVue user interface should be displayed.

You should be able to launch the online help files from the AutoVue client by selecting **Help > Online Help** from the AutoVue user interface.

Troubleshooting AutoVue Client

If you are unable to load the AutoVue client or the client starts up but is unable to connect to the AutoVue server, refer to following trouble-shooting pointers:

- Verify that all the proper JAR files are copied over to the Web server/application server, and that they match with the ones on the AutoVue server. This step is very important when upgrading your version of AutoVue.
- If you cannot load the HTML/JSP pages, ensure that you have the correct URL to the AutoVue client in the Web server/application server.
- If you are able to load the HTML/JSP pages, but the AutoVue client does not load:
 - Ensure that a JRE is installed on the client machine. If using a browser, ensure that the Java plugin is enabled with the browser.
 - Verify that the CODEBASE parameter set in the AutoVue client pages points to the correct URL.
- If you get an error message that indicates that there was a problem communicating with the server:
 - Verify that the JVUESERVER parameter in the client pages point to the right VueServlet URL.
 - Verify that the application server hosting the VueServlet is running.
 - Verify that the AutoVue server is running.

Installing in an Integrated Environment

This section describes the components that must be installed and configured when AutoVue is integrated with a DMS. Whether it is an Oracle VueLink integration or a custom integration with AutoVue, you must install and configure AutoVue client components and the VueServlet. The following are generic instructions for deploying AutoVue components in an integrated environment. For specific instructions refer to your VueLink/integration documentation.

Installing AutoVue Client Components in an Integrated Environment

In most cases, when AutoVue is integrated with a DMS, the AutoVue client components are deployed with an application server that hosts the integration components and/or the DMS.

In order to deploy the AutoVue client components in these environments, follow these steps:

- 1 Identify where the AutoVue client components are located in your integration/VueLink deployment.
- 2 If the AutoVue client components are deployed in a WAR file, extract the contents of the file.
- 3 Replace the following files in your integration environment (or extracted files) with files from the <AutoVue Install Root>\html directory:
 - jvue.jar
 - jogl.jar
 - gluegen-rt.jar
 - Online help files
- 4 If you want to support an offline deployment of AutoVue, you must copy *InstallDesktopDeployment.exe* (the installer for the Desktop Deployment) from your extracted Media Pack and *autovueupdate.xml* from the <AutoVue Install Root>\html directory to the same location as the AutoVue client components.
- 5 Specify the full URL to InstallDesktopDeployment.exe in the autovueupdate.xml file.
- 6 Re-create the WAR file (if you had extracted it in step 3).
Note: For information on creating a WAR file, refer to your VueLink/integration documentation.
- 7 Redeploy the WAR file.
- 8 Update jvueserver.properties and set the URL to the online help files to the URL specified by the application server.
Note: If you are using an application server cluster and a load balancer, it is recommended that you point to the load balancer address for the Online Help and the JAR files. This ensures backup for the online help and the JAR files in case of a failure of an application server instance.

Installing the VueServlet in an Integrated Environment

In most cases, when AutoVue is integrated with a DMS, the VueServlet is deployed on the application server that hosts the integration servlet and/or the DMS.

In order to deploy the VueServlet in these environments, follow these steps:

- 1 Identify where the VueServlet.jar is located in your integration/VueLink deployment.
- 2 If the VueServlet is deployed in a WAR file, extract the contents of the WAR file.

- 3 Replace VueServlet.jar from the extract with the VueServlet.jar file from the current release.
- 4 Modify any of the configuration parameters for the VueServlet as needed. Refer to section ["VueServlet Configuration Options"](#) for more information.
- 5 Re-create the WAR file (if you had to extract it in step 3).
- 6 Redeploy the WAR file.

Verifying your Integration

After you AutoVue client components and the VueServlet, you must verify that your integration works correctly with this version of AutoVue:

- Verify that you can load your DMS files in AutoVue.
- Verify Help-About from the AutoVue client to make sure you have the JAR files of the right version and build.
- Access the VueServlet page and verify that it reports the correct build number.
- Verify that you can launch the AutoVue online help files successfully.
- Verify the other features supported by your integration to make sure all features work as expected.

Configuring AutoVue Server Farm for High Usage

In order to meet your concurrent usage requirements, it may be necessary to setup more than one AutoVue server and balance requests to AutoVue across these servers. Configuring multiple AutoVue servers to communicate with each other to handle the load is referred to as an AutoVue server farm. Each AutoVue server has a primary server and multiple document servers. The primary server accepts all requests to AutoVue and is responsible for distributing document requests across the document servers. When AutoVue is configured in a server farm, the primary servers across the servers in the farm communicate with each other in order to distribute load across all the document servers in the server farm.

Setting Up AutoVue Server Load Balancing

- 1 Add a new machine to the same network as the original AutoVue server.
- 2 Install and configure the AutoVue server on the new server, going through the same steps as in the original installation.
Note: All the AutoVue servers in the cluster must be of the same version. If you install a patch or a service pack on one of the servers, you must do same for all the other servers.
- 3 Once installed, edit the `javueserver.properties` file located in the `<AutoVue Install Root>\bin` directory on the machines hosting the servers in the server farm and add the following parameters:

```
jvueserver.rmi.host.1=jvueserver1.company.com:1099
jvueserver.rmi.host.2=jvueserver2.company.com:1099
.
.
.
```

Where

`jvueserver.rmi.host.1` is set to the name and the RMI port of the one of the AutoVue servers in the farm, `jvueserver.rmi.host.2` is set to another AutoVue server in the farm and so on.

Note: Ensure that the RMI host entries are specified in the same order on all the servers in the server farm.

Symbol Libraries

If you are using symbol (stamp) markup entities, we recommend that you do not share the symbols folder between the servers in the farm. Instead, replicate the symbols folder across all AutoVue servers in the server farm at regular intervals.

Verifying AutoVue Server Load Balancing

In order to verify AutoVue server load balancing, you must:

- 1 Open a few different connections to the AutoVue server. Monitor the AutoVue server console on one of the servers in the farm. You should see that connections are being balanced across the servers in the server farm.
- 2 From these connections, open multiple documents. On the AutoVue server console, for each session, click on the Documents column and verify the server where the document is being opened from. The document requests should be load balanced across all the servers in the farm. Refer to section ["AutoVue Server Console"](#) for more information on how to see session and document information from the AutoVue server console.

Note:

- When opening a file using the upload:// protocol, the document is opened on the same server as the user's session.
- When opening a file, the DocServer with the least number of documents is selected. However, if two or more DocServers have the same load, then the DocServer that is on the same server as the session is selected.

Troubleshooting AutoVue Server Load Balancing

If you see that requests are not load balanced across the servers in the farm, verify the following:

- jvueserver.rmi.host.X entries are in the same order across all the servers in the farm
- All the servers in the farm are up and running
- A firewall is enabled on the machines where the AutoVue server is running. If a firewall is enabled, you must add java.exe and javaw.exe to the firewall exceptions.

Note: In some instances when you open connections simultaneously, it is possible that requests are not load balanced. This is as expected. When there is some lag between the connections, requests are load balanced.

Configuring VueServlet Load Balancing

The VueServlet needs to be deployed within an application server. You must rely on the load balancing capabilities of the application server or rely on an external load balancer that is configured to distribute load across all your application server (VueServlet) instances. You must also ensure that the load balancer is configured to enable session stickiness (also referred to as session persistence). Session stickiness is normally achieved through the use of browser cookies.

Ensure that each VueServlet instance has the same entry for the JVUESERVER parameter.

For example: If JVUESERVER=AVServer1:5099;AVServer2:5099 for one instance of the VueServlet, you must ensure that all the other VueServlet instances also specify the AutoVue servers in the same order.

Failover and Disaster Recovery

The following sections describe how to configure AutoVue for fail-over and disaster recovery.

AutoVue Server Configuration for Failover

For failover, AutoVue server should be deployed in a horizontal cluster or server farm. In a horizontal cluster, servers are spread over multiple machines.

You can have these AutoVue servers configured for load balancing so that if one AutoVue server in the server farm goes down requests are re-directed to other servers in the farm. Refer to section ["Configuring AutoVue Server Farm for High Usage"](#) for instructions on setting up a server farm. All AutoVue servers should be identified as peer servers acting as multiple entry points for all VueServlets communications (that is, there is no primary AutoVue server handling all the VueServlet communication). Each server in the farm acts as a backup server so that if one server goes down, another server is available to continue serving clients.

In certain situations, when you want a backup server without setting up load balancing, you must configure the VueServlet to communicate to the backup server if the production server is unavailable.

Note: When a server goes down, the users on that machine, along with all their open documents, are moved over to another machine. Any markups not stored in a DMS, or any user specific settings, are not moved over to the backup machine.

AutoVue Failover Configuration on the VueServlet

In the event of a failure of an AutoVue server, either when using a cluster or when using a standalone server, you can configure the VueServlet so that it directs requests to another AutoVue server. When using a cluster, the failover server can be another server in the cluster. When using standalone installation, you must install another instance of the AutoVue server.

To configure VueServlet for failover, update the JVUESERVER parameter of the VueServlet to add multiple AutoVue servers. Separate values using a ‘;’.

Note: Each VueServlet must have the same list of servers for the JVUESERVER parameter, and this list must be in the same order for all VueServlets.

```
<param-name>JVueServer</param-name>
<param-value>AutoVueServer1:5099;AutoVueServer2:5099;AutoVueServer3:5099</param-value>
```

Failover for the VueServlet

Since the VueServlet is hosted within an application server, you can rely on the application server's load balancing and high availability features. You must ensure that there are multiple VueServlet instances on separate machines so that if one instance of the VueServlet is not accessible, users are automatically redirected to another VueServlet instance.

Similarly, you must plan for backups for the AutoVue client components and online help within your application server so that if one instance is not available, users are redirected to another instance.

Failover for AutoVue client components

If the AutoVue client components are deployed with an application server, you can rely on the application server's load balancing and high availability features. Ensure that the JAR files and the online help files are served through the load balancer.

Verifying Failover Configuration

To verify that AutoVue is configured fully for failover, you must bring down certain nodes and verify that users are still able to connect and use AutoVue:

- Shutdown an AutoVue server. Preferably the server that is the entry point (the first AutoVue server in the VueServlet configuration) for AutoVue requests. Open connections to AutoVue and verify that users are connected to the backup servers.
- Bring down an application server instance hosting the VueServlet and the client components. Ensure that users are still able to launch the AutoVue client, load files and load online help files.

Integrating With a DMS

This section describes the additional configuration to consider when AutoVue is integrated with a DMS. Configuring AutoVue for multiple backend systems, creating intellistamps templates and backward compatibility options are described in this section.

Backward Compatibility Options

If you are using an older integration with AutoVue and you have not updated your integration to be compatible with version 20.x, AutoVue provides an option to allow for backward compatibility. To allow for backward compatibility for 19.3 VueLinks/integrations, you must add the following parameter in `javueserver.properties` (by default, this option is turned off):

```
dms.vuelink.version = [19.3]
```

Note: This option will be desupported in an upcoming release of AutoVue. We recommend that you upgrade your integration so that it is fully compatible with AutoVue 20.x.

Multiple Document Repositories

If AutoVue is integrated with multiple DMS, AutoVue's Universal File Chooser (File Open) dialog allows you to browse and search them. You can search/browse DMS even if your client has not already established a connection to the DMS. To enable AutoVue to browse/search when you have not yet established a connection with the DMS, you must create a file named `vuelinks.xml` in the `<AutoVue Install Root>\bin` directory with the following format:

```
<DMSList>
  <vuelink url="vuelink_url">
    <name>your_DMS_name</name>
    <DMSArgs>
      <DMSArg name="your_argument" value="your_value" />
    </DMSArgs>
    <seed>seed_url</seed>
  </vuelink>
</DMSList>
```

- The `<vuelink>` tag defines the URL location of the backend DMS system. Replace `vuelink_url` with the URL to your VueLink/integration servlet.
- The `<name>` tag defines the DMS button name to appear in the File Open dialog. Replace `your_DMS_name` with the name of your DMS.
- The `<DMSArgs>` tag defines arguments for the specified integration. Replace `your_argument` and `your_value` with any DMSArgs you may use with DMS integration.
- The `<seed>` tag defines the URL format for retrieving a file from the DMS. Replace `seed_url` with a URL to a file from your DMS. This is generally the FILENAME URL that is passed to the AutoVue client when you view your DMS file in AutoVue.

Note: In your seed URL, you must replace any special characters with it's character entity reference. For example: Replace `&` with `&`

For more information on the File Open dialog, refer to the *User's Manual*.

Note: If you wish to use this feature (browse/search DMS when you have not yet established a connection with the DMS) with VueLink for Documentum 19.3.x, you will need a hotfix for the VueLink. Contact a Oracle Customer Support representative for a hotfix for your VueLink.

Creating an Intellistamp Template

Intellistamps are dynamic stamp entities that can retrieve user and document metadata from the DMS. Intellistamps can also update attributes from the Intellistamp to the document in the DMS. They are only available to end-users after they have been configured by the system administrator.

The following are steps to enable intellistamp support:

- 1 Design an Intellistamp template using Stamp Designer. Refer to section ["Designing an Intellistamp Template"](#).
- 2 Configure designed Intellistamp templates. Refer to section ["Configuring Intellistamp Templates"](#).
- 3 Configure your VueLink to use designed Intellistamp templates. Refer to section ["Configuring Intellistamp with Your Integration"](#).

Designing an Intellistamp Template

To design an Intellistamp, you must first identify the image to use as a background and the attributes to display on this background. AutoVue supports adding Windows Metafile (WMF), Enhanced Metafile (EMF) and Windows Bitmap files as background images for your intellistamp.

Note: It is recommended to use a WMF or an EMF as the background image for your Intellistamps. These formats have the advantage that they support transparency and are generally smaller in size as compared to Bitmap files.

To design an Intellistamp template, do the following:

- 1 Go to the <AutoVue Install Root>\bin directory and run stampdlg.exe.
The Design Stamp dialog appears.
- 2 Specify a name for the stamp in the Stamp Name field.
- 3 In the Image File section, browse to the WMF/EMF/BMP file that you want to set as your stamp background.
- 4 Specify the default font that you want to use for the Intellistamp. To do so, click **Font** and specify the font details. This is the font that is used when you create an Intellistamp. The font changes when you resize the Intellistamp or when you change the font from the markup toolbar.

Note: When an Intellistamp is created in the AutoVue workspace, AutoVue scales the font up/down depending on whether the stamp is drawn larger/smaller in relation to the original WMF.

The end-user must modify the font from the Markup toolbar as necessary.

- 5 In the Attributes section, click **Add** to add the attributes that you want to be displayed on the Intellistamp:
 - Map the attribute to a DMS attribute. Specify the name of the attribute as it appears in the DMS in the Name field.

Note: You can only have attributes in the Intellistamp that map to attributes in the DMS. You cannot have Intellistamp attributes that do not map to DMS attributes.

- To map an attribute to the pre-defined variables, \$user or \$date, select the value from the Values list. If the DMS attribute does not have a value, these variables may be used to set the default value for the attribute. *\$user* is the name of the current user (either the DMS user name or the operating system user name if AutoVue is not integrated with a DMS) and *\$date* is the system date.
- To have an attribute that is not displayed on the Intellistamp, but is accessible from the Intellistamp Edit dialog, select the **Hidden** check box.
- To set an attribute as read-only, select the **Read Only** check box.

- 6 In the Preview section, select and place each attribute as required in the Intellistamp. Resize the attributes as necessary.
- 7 Repeat steps 2 through 6 to design any additional Intellistamp templates.

Configuring Intellistamp Templates

When Intellistamp templates are designed, the Intellistamp INI file, dmstamps.ini, is updated with information regarding the Intellistamp. The default location of this INI file is <AutoVue Install Root>\bin.

The following table contains a description of the available INI options in dmstamps.ini:

Note: Options in the [Stamps] section of the file apply to all Intellistamps.

[Stamps]

INI Option	Description
NumStamps=<integer>	This option indicates the total number of Intellistamps that have been designed. Important: Do not update this option value. Example: NumStamps=12 Indicates that there are 12 intellistamps that have been configured.
AttributesNames=attribute1;attribute2;...	This option lists the attribute names that should appear in the Stamp Designer list. Separate multiple attributes using a semi-colon (;). Ensure that the last attribute has a semi-colon after it. By default, the Stamp Designer has 2 drop-down items: approved_by and date_issued. Example: AttributesNames=approved_by;date_issued;dm_approval_status;
ReadOnlyAttributes=attribute1;attribute2;...	Identifies the attributes that should appear read-only by default in the Stamp Designer. Separate multiple values with a semi-colon (;). Example: ReadOnlyAttributes=date_issued

Each Intellistamp must have a [Stamp/*n*] section in the INI file, where *n* is an integer starting from 0 and is an index for each defined Intellistamp. For example, when two Intellistamps are designed, there should be two sections [Stamp0] and [Stamp1] in the dmstamps.ini file.

Each section contains information pertaining to the Intellistamp represented by the section:

[Stamp/*n*]

INI Option	Description	Default
Name=name of stamp	Specifies the name of the Intellistamp. Example: Name=ReviewerStamp	
ImageFile=full path to image	Sets the full path to the background image for the Intellistamp. Example: ImageFile=C:\stamps\reviewbg.wmf	
ReadOnly=[0 1]	Specifies whether or not so that the Intellistamp is read-only. Set to 1 so that the Intellistamp is a read-only stamp. Example: ReadOnly=1	0

INI Option	Description	Default
NumAttributes=<integer>	Indicates the number of attributes associated with this stamp Example: NumAttributes=3 This indicates that stamp has 3 attributes.	
Font	Specifies the default font for the stamp. This is the font that is used by the Intellistamp at creation time. The end-user can also change the font and font size when creating an Intellistamp. Example: Font=Arial,16,400	
Isotropic	Specifies if the Intellistamp should be resized isotropically. When Isotropic is set to 1 , the stamp resizes uniformly in all directions. Example: Isotropic=0	1
Attribute[n]	Specifies the name of the [n]th attribute Example: Attribute1=dm_approval_status	
Value[n]=[\$user \$date]	Specifies Value of the [n]th attribute Example: Value1=	
Format[n]	If value[n] is \$date , specify the format for the date. Example: Format1=EEE, MMM d, 'yy Any date format specified by java.text.SimpleDateFormat can be specified: http://download-l1nw.oracle.com/javase/6/docs/api/java/text/SimpleDateFormat.html	
ReadOnly[n]	Specifies whether or not Attribute[n] is a read-only attribute. If 1 , Attribute[n] is a read-only attribute Example: ReadOnly1=0	
Hidden[n]	Specifies whether or not Attribute[n] is hidden. If set to 1 , Attribute[n] is a hidden attribute Example: Hidden=0	
PosX[n] PosY[n]	Specifies the position of Attribute[n] relative to the background image. The top-left corner is 0,0. Example: PosX1=0.621622 PosY1=0.029661	

Configuring Intellistamp with Your Integration

The default location of the Intellistamp INI file, dmstamps.ini, is located in the <AutoVue Install Root>\bin directory and must be made available to your integration/VueLink. Additionally, when Intellistamp templates are created, they refer to paths for the background images. These images must be made available to the VueLink. Perform the following steps to ensure that the VueLink can access Intellistamp templates:

- 1 Identify where you want to store dmstamps.ini. This INI file can be placed in a folder path that is accessible to the VueLink or can be checked into the DMS.
- 2 If you want to check-in dmstamps.ini and the stamp background images into your DMS, do the following:

- a. Ensure that all users have read permissions to the folder in the repository where you will check-in the INI and the stamp background images.
- b. Check-in dmstamps.ini into the repository.
- c. Update vuelink.properties to point it to the path to the dmstamps.ini file in the repository. Edit vuelink.properties in a text editor such as Notepad. Update the value for CSI_IntellistampDefLocation to point to the full path to dmstamps.ini.

Example: CSI_IntelliStampDefLocation=/System/dmstamps.ini

- d. Check-in all the WMFs into the repository.
- e. Edit dmstamps.ini and update the ImageFile option value for all Intellistamps templates to point to the stamp background images in the repository.

Example: ImageFile=/System/intelimage1.wmf

- f. Repeat steps c through e for all repositories. Ensure that dmstamps.ini and the WMFs are available at the same path in all the repositories.

3 To have the Intellistamp templates in Windows folder paths, follow these steps:

- a. Copy dmstamps.ini to a folder path accessible to the VueLink.
- b. Update vuelink.properties to point to the path to the dmstamps.ini file. Edit vuelink.properties in a text editor such as Notepad. Update the value for CSI_IntellistampDefLocation to point to the full path to dmstamps.ini.

Example: CSI_IntelliStampDefLocation=C:/stamps/dmstamps.ini

Note: You must use forward slash (/) for the path to the INI file.

- c. Copy the background images to the same location as the INI file.
- d. Edit dmstamps.ini and update the path to the stamp background images to reflect the location where you placed them.

Example: ImageFile=C:\stamps\Intelimage1.wmf

- e. Repeat steps c through e for all VueLink instances. It is not recommended to use a shared network location for dmstamps.ini and/or the WMFs.

Verifying Your Integration

- If you are using a 19.3-compatible integration, verify that your integration works properly with AutoVue 20.1. Verify that you can load your DMS files in AutoVue. Verify other functionality that is supported by your integration to ensure that your integration will work with AutoVue 20.1.
- If you are using Intellistamps, verify that users can create Intellistamps. Verify that the attributes in the Intellistamp reflect backend system attributes.
- If you are working with multiple DMSes and you have vuelinks.xml configured, launch AutoVue applet and verify that you can browse or search through the DMSes.

Offline/Disconnected Use of AutoVue

AutoVue in Online/Offline Mode

The offline mode option in AutoVue provides you with the ability to view and markup files when you do not have access to the AutoVue server or DMS. Many AutoVue users require access to their files when they are not connected to their DMS. For example, having offline access to files is essential when you need to review and add markups to a document during your regular commute or when bringing all your files on a business trip.

Note: Going offline is supported from Windows-based clients only.

When you select the **Work Offline** option in AutoVue, a predefined list of files and associated resources and markups are copied to your local system as offline files and a local installation of AutoVue is deployed. Once installation is complete you can continue working on your files with AutoVue in offline mode. For more information refer to the *Oracle AutoVue User's Manual*.

Note: If you are going offline from an HTTPS site that has a test or a trial certificate, you must add the certificate to the offline installation's trusted store using the keystore java utility. Note that if you have a valid certificate, you do not have to do this step.

Configuring for Offline Mode

- 1 Copy autovueupdate.xml from <AutoVue Install Root>/html to the Web server or the application server that you are using for AutoVue client components.
- 2 Download the AutoVue media pack for Windows and extract the InstallDesktopDeployment.exe for the Desktop deployment installation of AutoVue.
- 3 Copy this executable to the Web server or the application server location where AutoVue client files are deployed.
- 4 Edit autovueupdate.xml and specify the URL to the AutoVue executable.
Example: url='http://autovueserver1:80/AutoVue/InstallDesktopDeployment.exe'
- 5 Specify the file path to autovueupdate.xml in the jvueserver.update.xml.url parameter in jvueserver.properties. Refer to section ["Offline Mode Option"](#) for more information.

Enabling Markups for Office Formats in Offline Mode

When in offline mode, the Markup feature is disabled by default for Office formats. To enable the option, you must set the ENABLEOFFICEMARKUPS INI option to 1. For more information, refer to the "Viewing Configuration for Office Formats" section in the *Viewing Configuration Guide*.

AutoVue Mobile

AutoVue Mobile is an add-on product that can be installed on top of AutoVue Client/Server Deployment to enable viewing and marking up of design documents in a disconnected mode. AutoVue Mobile enables users to share design documents and markups with their partners or suppliers that do not have access to their document management system. Using the AutoVue Mobile feature in AutoVue, you can create a Mobile Pack (a "packaged" file that contains the base file, all the external resources—fonts, XRefs—needed to fully display the file, and existing markups for the file). Once the Mobile Pack is created, they can provide it to their suppliers/partners who can use Oracle AutoVue Desktop Deployment in order to view and markup the file in the Mobile Pack.

Installing AutoVue Mobile

Note: You must shutdown AutoVue before you install AutoVue Mobile.

To install AutoVue Mobile:

- 1 Download the Oracle AutoVue Media pack and extract its contents.
- 1 From the extracted Oracle AutoVue Media Pack, run the AutoVue Mobile executable:
Windows OS: The installer is avmsetup.exe
Linux OS: The installer is avmsetup.bin.
Note: You might need to grant execute permissions for the installer on Linux. To do so, run `chmod +x avmsetup.bin`.
- 2 Select a language from the installation dialog and then click **OK**.
- 3 Click **Next** to begin installation.
- 4 Review the pre-installation summary and then click **Install**.
Note: AutoVue Mobile is automatically installed in the same directory as AutoVue. For information on installing AutoVue, refer to section ["Basic AutoVue Server Installation"](#).
- 5 When installation is complete, click **Done** to close the installer.

Defining Markup Policy

When working with Mobile Packs, the markup permissions on the Mobile Pack is determined by the markup policy defined during Mobile Pack creation. Markup policy defines a set of rules to determine certain restrictions and privileges for users of the Mobile Pack. If no markup policy is defined in the Mobile Pack, a default set of values are used.

The default markup policy file, MarkupPolicy.xml, can be found in the <AutoVue Install Root>\bin folder.

Note: For more information on Mobile Pack, refer to the *User's Manual*.

The following table explains the actions included in the markup policy, their default values, and which values can be modified.

All the following actions can be combined with certain exceptions.

Action	Description	Default
SaveNewMarkup= <TRUE FALSE>	If TRUE, new Markup file creation inside the Mobile Pack is allowed. If FALSE, new Markup file creation inside the Mobile Pack is not allowed: For new/imported (local file) Markup file, Markup Save and Markup Save As are disabled. For an existing Markup file, Markup Save As is disabled.	TRUE
SaveExistingMarkup= <TRUE FALSE>	If TRUE, resave of existing Markup file is allowed. If FALSE, the Markup file from the Mobile Pack cannot be re-saved. The Markup Save command will be disabled for the existing Markup file.	TRUE
EditMarkup= <TRUE FALSE>	If TRUE, the Markup file can be edited. If FALSE, the Markup file is opened as read-only. All manipulation commands at all levels—Markup file, markup layers, and markup entities—are disabled. Setting EditMarkup=False is equivalent to the Hide Icon menu option.	TRUE

Action	Description	Default
DeleteMarkup= <TRUE FALSE>	<p>Note: Only available for a Mobile Pack created with DMS files.</p> <p>If TRUE, the Delete Markup menu option is enabled in the Markup File Open dialog.</p> <p>If FALSE, the Delete Markup menu option is disabled in the Markup File Open dialog.</p>	TRUE
OpenMarkup= <TRUE FALSE>	<p>If TRUE, the Markup file is listed in the Markup File Open dialog.</p> <p>If FALSE, the Markup files is not listed in the Markup File Open dialog.</p>	TRUE
AutoOpenMarkup= <TRUE FALSE>	<p>If TRUE, the Markup file opens automatically when opening the Mobile Pack.</p> <p>If FALSE, the Markup file does not open automatically when opening the Mobile Pack.</p>	FALSE
FilterAttrFromGUI.<GUI>.<Prop>.<Value>=<TRUE FALSE>	<p>If TRUE, the given property value is removed from the GUI definition.</p> <p>If FALSE, the given property value is not removed from the GUI definition.</p> <p>This action takes three parameters:</p> <p>GUI: GUI definition to be modified</p> <p>Prop: Property to be removed</p> <p>Value: Property value to be removed</p> <p>If no Value is specified, then the UI element that represents the given property in the GUI definition is removed.</p> <p>For example:</p> <p><Action name="FilterAttrFromGUI.Edit.CSI_MarkupType.master" default="false"></p>	FALSE

The following is an example of a markup policy.

```
<MarkupPolicy>
<Action name="SaveExistingMarkup" default="true">
<ExConditions>
<OrOperator>
<AndOperator>
<AnyMarkupFileCondition Name="CSI_MarkupType" Value="master"/>
<MarkupFileCondition name="CSI_MarkupType" value="consolidated"/>
<MarkupFileCondition name="CSI_DocAuthor" value="$CURRENT_USER"/>
</AndOperator>
<NotOperator>
<MarkupFileCondition name="CSI_DocAuthor" value="$CURRENT_USER"/>
</NotOperator>
</OrOperator>
</ExConditions>
</Action>
<Action name="EditMarkup" default="true">
<ExConditions>
MarkupFileCondition name="Original" value="true"/>
</ExConditions>
</Action>
<Action name="DeleteMarkup" default="true">
<ExConditions>
<OrOperator>
<AndOperator>
<AnyMarkupFileCondition name="CSI_MarkupType" value="master"/>
<MarkupFileCondition name="CSI_MarkupType" value="consolidated"/>
<MarkupFileCondition name="CSI_DocAuthor" value="$CURRENT_USER"/>
</AndOperator>
<NotOperator>
<MarkupFileCondition name="CSI_DocAuthor" value="$CURRENT_USER"/>
</NotOperator>
</OrOperator>
</ExConditions>
</Action>
<Action name="FilterAttrFromGUI.Edit.CSI_MarkupType.master"
default="false">
<ExConditions>
<AnyMarkupFileCondition name="CSI_MarkupType" value="master"/>
</ExConditions>
</Action> </MarkupPolicy>
```

Configuring for Real-Time Collaboration

The following section describes how to configure AutoVue for Real-Time Collaboration. All configurations are performed in `javueserver.properties`. For a complete list of AutoVue server configuration options including collaboration options, refer to ["AutoVue Server Configuration Options"](#).

Default Collaboration Configuration

When configuring AutoVue for a real-time collaboration deployment, you must set the following basic parameters in `javueserver.properties`.

- `javueserver.collaboration.enable`
This parameter must be set to TRUE to enable collaboration mode on the AutoVue server. By default, this parameter is set to TRUE.
- `javueserver.collaboration.protocol`
You must specify if you want to use RMI or JXTA for collaboration when AutoVue server is deployed in a server farm. The default is RMI and is the recommended protocol. JXTA should be used when the servers in the server farm are separated by more than one firewall.
- `javueserver.collaboration.tcp.port`
If you have setup the AutoVue servers in a server farm, you must specify the TCP port to use. This port is used by JXTA (when the collaboration protocol is set to JXTA) for communication between the servers in the farm.
- `javueserver.collaboration.id.min`
When running an AutoVue server farm, you can specify the minimum ID to use for collaboration sessions and users by this AutoVue server. The second server in the farm should have a minimum ID of at least the first server's minimum ID + the first server's ID range.
- `javueserver.collaboration.id.range`
Specify the range of IDs given to collaboration and users by this AutoVue server.
- `javueserver.collaboration.group`
Specify the collaboration group to which an AutoVue server belongs. This is required when JXTA is used for collaboration.

Refer to section ["Collaboration Options"](#) for more information.

You must perform additional configurations if you have multiple AutoVue servers in server farm or behind a firewall. For more information, refer to section ["Configuring Across Multiple Firewalls and AutoVue Servers"](#).

Distributed Geographies Configuration

Configuring for distributed geographies consists of the same steps as described in section ["Default Collaboration Configuration"](#).

Distributed DMS Configuration

Configuring for distributed DMS consists of the same steps as described in section ["Default Collaboration Configuration"](#), except that there must also be an internal virtual private network (VPN) setup between the AutoVue servers.

Configuring Across Multiple Firewalls and AutoVue Servers

If you have more than one firewall separating the AutoVue servers that you want to use for collaboration, you must rely on JXTA for communication between the servers.

Note: JXTA configuration is only available when AutoVue server is running on Windows platforms.

In addition to the parameters set in section ["Default Collaboration Configuration"](#), you must configure additional parameters when AutoVue servers in a server farm are behind a firewall.

The following table lists the configurable parameters for JXTA configuration.

Parameter	Description	Default
<code>jvueserver.collaboration.protocol=[jxta]</code>	Protocol should be set to JXTA.	
<code>jvueserver.collaboration.rendezvous.enable=[TRUE FALSE]</code>	Set to true to enable communication with other servers that are not part of the server farm. When you have multiple server farms, set to true for at least one server in each farm to enable this server to communicate with other server farms across firewalls.	
<code>jvueserver.collaboration.rendezvous=[protocol://IP_of_server_to_communicate_with:port]</code>	Specify the protocol, the IP address of other servers to communicate with, and the port for communication. For example: <code>jvueserver.collaboration.rendezvous=tcp://ip1:port1;http://ip2:port2</code>	
<code>jvueserver.collaboration.jxta.allowExternal=[TRUE FALSE]</code>	Set to true to allow other servers that are not part of the server farm to communicate with this server.	

Specify one of the following parameters when using network address translators in a firewall setup.

Parameter	Description	Default
<code>jvueserver.collaboration.http.server=[external_IP:port]</code>	When using firewalls and Network Address Translators, specify the external address and port for http connections.	
<code>jvueserver.collaboration.tcp.server=[external_IP:port]</code>	When using firewalls and Network Address Translators, specify the external address and port for TCP connections.	

The following example describes how to configure two AutoVue servers (Server A and Server B) for collaboration with the JXTA protocol across a wide area network (WAN) using HTTP protocol. Note that there must be an open HTTP port on the firewall that allows bi-directional travel of information.

This example assumes Server A has IP address `aaa.aaa.aaa.aaa` and Server B has IP address `bbb.bbb.bbb.bbb`. If you are using a DNS service (which is the recommended approach), you should include the host name instead.

Alternately, you may configure port forwarding on the firewall to point to a standard port (for example, 80), and have the firewall recognize the requestor and then forward to the AutoVue server.

For each machine hosting the AutoVue server, you must modify the Collaboration Settings in each `jvueserver.properties` file. For example, the following code sample shows the modification for the `jvueserver.properties` file on Server A:

```
#
# * Enable or disable collaboration
jvueserver.collaboration.enable=true

# * Communication protocol
jvueserver.collaboration.protocol=jxta

# * Minimum of IDs for collaboration
jvueserver.collaboration.id.min=0

# * Range of IDs for each server
jvueserver.collaboration.id.range=100000

# * Base tcp port for collaboration
jvueserver.collaboration.tcp.port=9700

# * Base http port for collaboration
jvueserver.collaboration.http.port=9800

# * Group name
jvueserver.collaboration.group=Oracle

#enable communication with other servers that are not part of the server
farm
jvueserver.collaboration.rendezvous.enable=true

#IP and port of other servers to communicate with
jvueserver.collaboration.rendezvous=http://bbb.bbb.bbb.bbb:9800

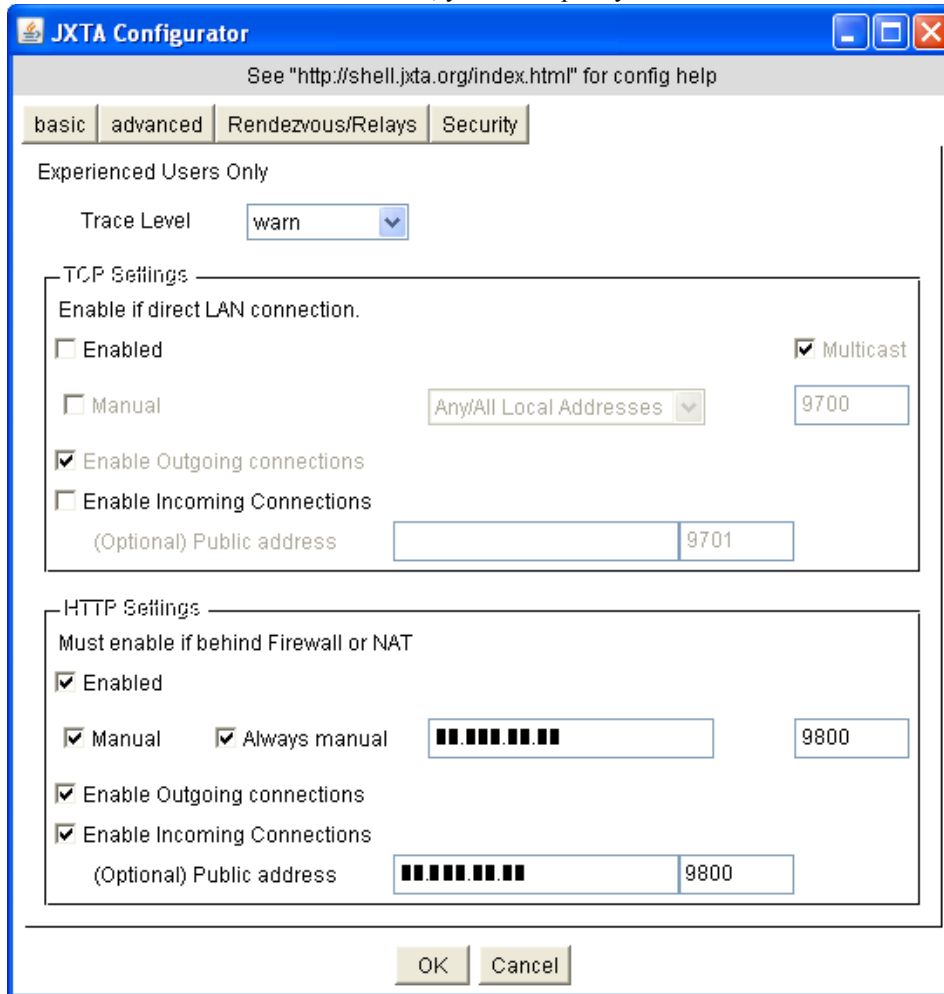
# allow servers not part of the server farm to communicate with this
server
jvueserver.collaboration.jxta.allowExternal=true

jvueserver.collaboration.config.manual=true

#When using firewalls and NATs, this is the external address and port for
communications
jvueserver.collaboration.http.server=aaa.aaa.aaa.aaa:9800
```

The JXTA Configuration dialog appears when the changes are made and either server is restarted.

Note: In this example, TCP communication is disabled and only HTTP protocol is used. However, TCP can also be used. By default, `juveserver.properties` uses port 9800 for HTTP and port 9700 for TCP. In this screenshot, the IP addresses have been blacked out; you must specify valid IP address or hostnames.



- 1 When the AutoVue servers have restarted, make sure that the `VueServlet` is running.
- 2 Start a real-time collaboration session. Refer to the *Oracle AutoVue User's Manual* for more information.

Note: You must use a file that is visible to both AutoVue servers. For example, use a file that can be located via a Web URL.

Both clients should be visible when you start the collaboration session.

Note: The performance of the collaboration depends on typical network factors such as quality connection, geographic distance, and so on.

Starting the AutoVue Server

This chapter discusses how to start and stop the AutoVue server on Windows and Linux.

Starting AutoVue on Windows

- 1 Start the AutoVue server by clicking **Start AutoVue Server**  in the **Oracle AutoVue** Program Manager group.

By default, when the server is started, the console is displayed and the server appears in the system tray.

When you start the server as a service, you may not see the server console. To display the console, run the following command from the <AutoVue Install Root>\bin directory:

```
jvueserver_debug -u
```

Note: The AutoVue server starts up with a default ProcessPoolSize of 4. To modify the ProcessPoolSize, set the `jvueserver.processPoolSize` parameter in `jvueserver.properties`. Refer to section ["AutoVue Server Configuration Options"](#) in for more information.

- 2 Start the application server on which `VueServlet` is deployed.

Note: If you are using Jetty, you must start it up by running the **Start VueServlet on Jetty** shortcut in the AutoVue programs group.

- 3 Make sure to start the Web server if you are using it for the AutoVue client components.

Note: For information on starting AutoVue as a service, refer to section ["Running the AutoVue Server as a Service"](#).

Starting AutoVue on Linux

- 1 Start the AutoVue server by entering the following:

```
./jvueserver
```

This starts up the server console as long as the `DISPLAY` environment variable is properly set.

When you start the server as a service, or when the `DISPLAY` environment variable is not set properly, you will not see the server console. To display the console, run the following command from the <AutoVue Install Root>/bin directory:

```
./jvueserver_debug -u
```

Note: The AutoVue server starts up with a default ProcessPoolSize of 4. To modify the ProcessPoolSize, set the `jvueserver.processPoolSize` parameter in `jvueserver.properties`. Refer to section ["AutoVue Server Configuration Options"](#) for more information.

- 2 Start the application server on which `VueServlet` is deployed.

Note: If you are using Jetty, you must start it up by running `startJetty` from the <AutoVue Install Root>/bin/jetty/bin directory.

- 3 Make sure to start the Web server, if you are using it, for AutoVue client components.

The startup script for the AutoVue server on Linux OSes also starts up the Xvfb server. Xvfb is an X11 virtual framebuffer that helps the AutoVue server render files. The Xvfb server runs on port 909 by default. To modify this

port and configure other Xvfb properties, open `javueserver.properties` (located in the `<AutoVue Install Root>/bin` directory) and locate property names containing “xvfb”.

If you want the AutoVue server to continue running after you close the terminal window, or after you log out of the Linux machine, you must exit the shell (console window) used to start the AutoVue server before logging out of Linux. The server continues running even after you log off. To exit the shell, you must enter `Exit` (do not exit by clicking the Close button).

Note: For information on starting AutoVue as a service, refer to section ["Running the AutoVue Server as a Service"](#).

Shutting Down the AutoVue Server

To shut down the AutoVue server, click **Shutdown** on the AutoVue server console. If you are running the AutoVue server as a service, you must shut it down as you would any service.

Running the AutoVue Server as a Service

When running the AutoVue server as a service, you must run it as a *named user* and not as Local System Account, as the local system account has more privileges than a named account.

On Windows OSes

AutoVue server can be run as a Windows Service. The advantage of this is that it continues to run even after you log off of Windows. Before running the AutoVue service, first verify that it runs properly in “non-service” mode (for example, run by clicking the **Start AutoVue Server** button in the **Start** menu).

To install the service, go to the `\bin` folder of the directory where you installed the AutoVue server and enter the following:

```
javueserverX.exe -install <user information>
```

where `<user information>` is in the form “domain\username password”. This ensures that the AutoVue Server service runs as a named user instead of the local system account.

To remove the service, go to the `<AutoVue Install Root>\bin` directory and enter the following:

```
javueserverX.exe -remove
```

Starting and Stopping the Service

- 1 In the Control Panel start **Services**.
- 2 Select the **Oracle AutoVue Server** service.
- 3 Click **Startup**.
- 4 Select whether you want the service started automatically on re-boot or manually. The default is **Manual**.
- 5 If you select **Manual**, you can start the service by doing one of the following:
 - Click **Start** in the Services dialog
 - or
 - Use the `sc.exe` utility.**For example:** `SC start "Oracle AutoVue Server"`

or

- Use the NET program.

For example: NET start "Oracle AutoVue Server"

Once the Service has been started, it behaves exactly as if running in “non-service” mode. The AutoVue server icon appears in the System Tray. To stop the service click **Shutdown**.

On Linux OSes

Oracle provides an *RC-Script* to manage the AutoVue server on Linux. The AutoVue server can be configured to startup automatically when the machine is restarted by following these steps:

- 1 Edit file *<AutoVue Install Root>/etc/jvueserver_rc* and locate the following lines:

```
AUTOVUEDIR=$USER_INSTALL_DIR$
```

```
AUTOVUEUSER=__JVUEUSER__
```

- 2 Replace `$USER_INSTALL_DIR$` with the path to AutoVue installation and `__JVUEUSER__` with the name of the user that will be running the AutoVue server.

- 3 Rename *jvueserver_rc* to *autovue*.

- 4 Login as a root and copy *autovue* to */etc/init.d* folder.

- 5 As root, go to the */etc/init.d* folder and add AutoVue as a service:

```
chkconfig --add autovue
```

- 6 Configure *autovue* to startup automatically:

```
chkconfig autovue on
```

AutoVue now starts up automatically when the machine starts up.

To start the AutoVue service, manually, run

```
service autovue start
```

To stop the AutoVue service manually, run

```
service autovue stop
```

To remove the AutoVue service, run

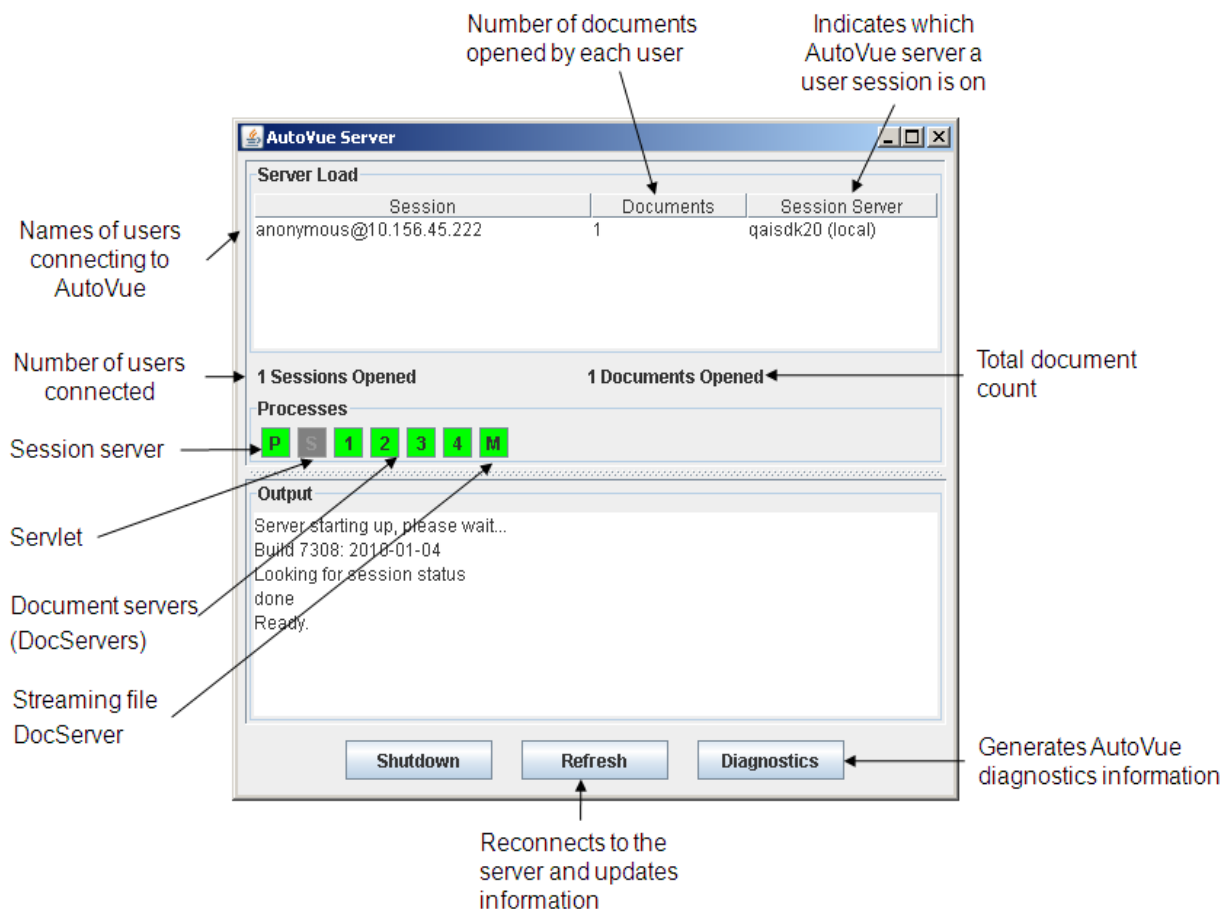
```
chkconfig -del AutoVue
```

Monitoring the AutoVue Server

You can monitor the AutoVue server from the AutoVue Server Console. The console displays information on the number of clients connected to AutoVue, the document opened by each connection, the server that a user session is connected to. Additional information such as AutoVue usage history, server diagnostics are also available. The following sections describe how to go about monitoring your AutoVue server.






AutoVue Server Console








The AutoVue server console displays the user connection state (process, username, client IP and number of open documents) and the process pool state. Upon starting the server, the console launches and the connection and process pool states are queried.



- Click **Refresh** to update the console display to regenerate server information.
- To stop running the AutoVue server and all attached processes, click **Shutdown**.
- The **Diagnostics** feature of the AutoVue server console generates a report, *JVueServerDiagnostics.out*, to the <AutoVue Install Root>\bin\logs directory and contains pertinent troubleshooting information. An Oracle Global Customer Support representative may require you to generate the report to identify problems you may have with your deployment of AutoVue. Any errors that occur during initialization are listed under **Output**.

- The Processes section of the console lists the servers and their status:

Pool State	Description
 Red	Process is not running.
 Green	Process is running.
 Yellow	Process is initializing.
 Grey	Process is disabled by the user (applies only to servlet process).
 Black	Process is not responding.

-  indicates the session server. You can view AutoVue server information (build number and server load) by clicking on the  button.
-  indicates the servlet engine.
- , ,  ... represent document servers (DocServers). You can restart a DocServer by clicking on the button and selecting **Restart**. The number of document servers is set in the `javueserverx.nt.processPoolSize` parameter in `javueserver.properties`. For more information, refer to section ["Process Pool Size Option"](#).
-  represents an additional DocServer reserved for generating streaming files (only visible when `javueserver.metacache.process` is set to **TRUE**, which is the default value in `javueserver.properties`). You can restart this server by clicking on the button and selecting **Restart**.
- Session information is displayed on the Console:
 - The names of the users connecting to AutoVue.
 - The number of documents opened by each user.
 - The AutoVue server that the user session is on.
- Double-click on the session listed in the Console to view the following information regarding the session:
 - Which document(s) are currently opened by the user.
 - Which DocServer(s) are loading the user's documents.
 - Which AutoVue server(s) are loading the user's documents.

Usage Monitoring

AutoVue has usage monitoring to enable system administrators to track how many files of a format group are opened at any given time. For example, you can use this feature to track the number of licenses for the different product variations of a single deployment of AutoVue. Usage data is written to *licusage.out* file in the <AutoVue Install Root>\bin\logs directory.

AutoVue ships a utility to parse the usage log and present meaningful information to the system administrator.

The following is the format of the command line to run this utility:

```
usagestat [-c] <path to the input file>
```

where `usagestat` is the command to run this utility.

`[-c]` is an optional parameter and indicates that the utility should run in continuous mode (that is, the output displays continuously).

`<path to the input file>` specifies the full path to the input file (for example, the log file on which the statistics is based). This argument is mandatory.

The following is an example command line that runs the utility in a standard mode.

```
usagestat c:\AutoVue\bin\logs\LicUsage.out
```

Logging for the AutoVue Server

The configuration file **log4j.xml** (located in the <AutoVue Install Root>\bin directory) lets you configure the logging for the AutoVue server. By default, the logs are saved to the <AutoVue Install Root>\bin\logs directory. The configuration file defines several appenders (Log4j output destinations) and output layouts. Review the logging information in log4j.xml to troubleshoot any issues you experience with the AutoVue server. If you are unable to resolve the issue yourself, provide the logging information in log4j.xml to an Oracle Global Customer Support representative.

To set the logging level and time interval for detecting log4j configuration change, you must set log4j parameters in jvueserver.properties. For more information, refer to section ["log4j and Diagnostics Options"](#).

The following sections provide information on the available appenders, output layouts, and logger information.

Log4j Appenders

The following table lists the appenders that are defined in log4j.xml. However, you may use any other appender as you see fit.

Appender	Description	Parameters
File Appender (org.apache.log4j.FileAppender)	Appender for logging to file.	File: The name of the log file. Append: If set to <i>TRUE</i> , the logs are appended to the file after the process restarts. If set to <i>FALSE</i> , the old files are discarded after the process restarts.
Rolling File Appender (org.apache.log4j.RollingFileAppender)	Backs up (rolls) previous files when the maximum files size is reached. Note: This appender is enabled by default.	MaxFileSize: The maximum size that the output file is allowed to reach before being rolled over to the backup files. MaxBackupIndex: The maximum number of backup files to keep. \${jvueserver.processIndex}: The AutoVue server-specific parameter used to log AutoVue's processes into a separate file.
Daily Rolling File Appender (org.apache.log4j.DailyRollingFileAppender)	Defines the frequency for rolling over a file.	DatePattern: Determines the roll-over schedule. For format details, refer to log4j documentation.
Console Appender (org.apache.log4j.ConsoleAppender)	Used for logging the program console window.	

Appender	Description	Parameters
SMTP Appender (org.apache.log4j.SMTPAppender)	Send an e-mail when a specific logging event occurs (such as errors or fatal errors).	<p>BufferSize: The maximum number of logging events to collect in a cyclic buffer.</p> <p>SMTPHost: The host name of the SMTP server that sends the e-mail.</p> <p>From: E-mail address of the sender.</p> <p>To: A comma-separated list of the recipients' e-mail addresses.</p> <p>Subject: The subject of the e-mail.</p>
NT Event Log Appender (org.apache.log4j.NTEventLogAppender)	<p>Appends to the NT event log system.</p> <p>Note: This appender is only for Windows OSes.</p>	<p>Source: The source of the events.</p>

Output Layout

The following table defines the available output layouts in the log4j.xml configuration file.

Note: The output XML file can be viewed in a GUI-based log viewer such as Apache Chainsaw.

Output Layout	Description	Parameters
XML Layout (org.apache.log4j.xml.XMLLayout)	When this layout is enabled, log4j outputs the logs in XML format.	<p>Properties: Set this value to TRUE to force log4j to record Mapped Diagnostic Complex (MDC) values.</p> <p>AutoVue-specific MDC values:</p> <p>User: Outputs source username with logging event.</p> <p>Document: Outputs current document name with logging event.</p>
Pattern Layout (com.cimmetry.jvueserver.logger.JVuePatternLayout)	When this layout is enabled, log4j outputs the logs in textual format allowing for flexible string format configuration.	<p>ConversionPattern: The string that controls formatting. For the list of formatting characters, refer to log4j documentation.</p> <p>AutoVue-specific formatting characters in conversion pattern:</p> <p>%s: Outputs current document server index or "0" in the case of a session server.</p>

Note: To enable logging output to the console, you must uncomment the `JVUE-CONS` line in the `log4j.xml` configuration file as shown in the following figure:

```
<!-- Root logger -->
<!-- Uncomment additional appender-ref to output to different/multiple
outputs -->
<root>
  <level value="debug"/>
  <appender-ref ref="JVUE-ROLL"/>
  <!-- <appender-ref ref="JVUE-FILE"/> -->
  <!-- <appender-ref ref="JVUE-DAILY"/> -->
  <b>appender-ref ref="JVUE-CONS"/>
  <!-- <appender-ref ref="JVUE-SMTP"/> -->
  <!-- <appender-ref ref="JVUE-EVENT"/> -->
</root>
```

Logger Information

The following descriptions explain what kind of logger information will be seen for each class specified:

Class	Description
<code>com.cimmetry.jvueserver.management</code>	Displays information relating the start-up of the AutoVue server, communications between the AutoVue server clusters and connections from the console, and other server management-related reports.
<code>com.cimmetry.jvueserver.usage</code>	Displays information related to the usage of the AutoVue server (opening and closing sessions and documents).
<code>com.cimmetry.jvueserver.configuration</code>	Displays reports on loading errors of the server's configuration.
<code>com.cimmetry.jvueserver.event</code>	Displays information concerning posting and handling of different server events (opened and closed sessions, opened and closed documents, and so on).
<code>com.cimmetry.jvueserver.cache</code>	Displays information concerning the server's cache. Reports messages and errors related to loading the cache, locking, saving, deleting cached files as well as searching for archive and XRef files.
<code>log4j.category.com.cimmetry.connection</code>	Displays information concerning downloading files from the network.
<code>com.cimmetry.jvueserver.session</code>	Displays reports on sessions opening, closing and being restored, and the loading and saving of session profiles.
<code>com.cimmetry.jvueserver.document</code>	Displays document-related information (open, information, properties, and so on).
<code>com.cimmetry.jvueserver.document.native</code>	Displays messages and error reporting for document related native code execution.
<code>com.cimmetry.jvueserver.dms</code>	Displays DMS-related operations (open, download, save, properties, and so on).
<code>com.cimmetry.jvueserver.mobilepack</code>	Displays information concerning generation and usage of mobile packs.

<code>com.cimmetry.jvueserver.streamingfile</code>	Displays information concerning generation and usage of streaming files.
<code>com.cimmetry.jvueserver.collaboration</code>	Displays all server-side collaboration activity.
<code>com.cimmetry.jvueserver.collaboration.jxta</code>	Displays all messages regarding JXTA connection handling for real-time collaboration.
<code>com.cimmetry.jvueserver.collaboration.rmi</code>	Displays all messages regarding RMI connection handling for real-time collaboration.
<code>com.cimmetry.jvueserver.console</code>	Displays messages on server console loading, connecting information and Server Console Frame errors.
<code>com.cimmetry.frontend</code>	Displays all messages and error reported from the AutoVue client.

You can specify what kind of information to output by setting the classes to one of the following information levels:

Information Level	Description
OFF	Turn off all logging.
FATAL	Logs severe events that could cause the application to abort.
ERROR	Logs error events that might still allow the application to continue running.
WARN	Logs potentially harmful situations. This is the default logging level.
INFO	Logs informational messages that highlight the progress of the application at coarse-grained level.
DEBUG	Logs fine-grained informational events that are most useful to debug an application.

Note: If you need more specific error messages, you can turn on verbosity for specific classes.

For Example:

```
com.cimmetry.jvueserver.management=INFO
com.cimmetry.jvueserver.session=WARN
com.cimmetry.jvueserver.document=ERROR
com.cimmetry.jvueserver.dms=FATAL
```

These four lines mean that informational messages will be logged for the management class, warning messages will display for the session class, error messages pertaining to document requests will display for the document class. For the `com.cimmetry.jvueserver.dms` package, fatal messages will be reported.

Refer to the Apache Web site and log4j documentation for more information.

Customizing the AutoVue Client

AutoVue allows you to customize the client applet and graphical user interface. For example, you can change the locale for the AutoVue client or you can customize the AutoVue user interface by modifying the menu and the toolbars.

The following sections describe in detail how to configure AutoVue to your needs.

AutoVue Applet Parameters

AutoVue allows you to customize the client applet. For example, with the `EMBEDDED` parameter you can embed the applet into a Web page, or with the `DMS` parameter you can specify the DMS servlet that the AutoVue server uses to interface with a DMS.

The following table describes the customizable parameters in the AutoVue applet.

Syntax:

`<PARAM NAME=<name> VALUE=<type> >`

Refer to ["Basic Applet"](#) for a sample applet definition.

Name	Type	Value
ALLOWEDFORMATCLASSES	[Office 2D 3D EDA]	<p>You can use this parameter to override the formats allowed by the server. For example, if the parameter is set to Office;2D, user will only be able to view Office and 2D formats. User will not be allowed to view EDA or 3D even if the server supports these formats.</p> <p>The value is a semicolon-separated list.</p> <p>For example: ALLOWEDFORMATCLASSES=Office;2D</p> <p>Note: This parameter can only be used to restrict the formats, not allow more formats than the server license allows.</p>
CACHEUI	[TRUE FALSE]	<p>Set to TRUE to cache UI components for later use during the same Java Virtual Machine session. This can be set to TRUE to have better startup times for the AutoVue applet when in situations where the applet is reloaded multiple times.</p> <p>Default: FALSE</p>
COLLABORATION		<p>The parameters and values described here are set automatically when initiating and joining collaboration sessions from the AutoVue client.</p>

Name	Type	Value
	INIT:	Initiate collaboration session.
	<ul style="list-style-type: none"> CSI_ClbSessionID= 987654321 	DMS collaboration session ID.
	<ul style="list-style-type: none"> CSI_ClbDMS=dmsIndex 	DMS index.
	<ul style="list-style-type: none"> CSI_ClbSessionData= 123456789 	DMS collaboration session data.
	<ul style="list-style-type: none"> CSI_ClbSessionSubject= Subject 	Collaboration session subject.
	<ul style="list-style-type: none"> CSI_ClbSessionType= public private 	Collaboration session type.
	<ul style="list-style-type: none"> CSI_ClbUsers=user1, user2,... 	Invited users.
	JOIN:	Join collaboration session in progress.
	<ul style="list-style-type: none"> CSI_ClbSessionID= 987654321 	DMS collaboration session ID.
	<ul style="list-style-type: none"> CSI_ClbDMS=dmsIndex 	DMS index.
	<ul style="list-style-type: none"> CSI_ClbSessionData= 123456789 	DMS collaboration session data.
DMS	<i>http://name:port/dmsServlet</i>	Specifies the DMS servlet that the AutoVue server uses to interface with a DMS.
DMSARGS	<i>String</i>	<p>List of DMS arguments passed in as Applet parameters. Specify semicolon separated list of applet parameters. The value will be sent with every request to the DMS.</p> <p>Example:</p> <pre><PARAM NAME="DMSARGS" VALUE="ARG1;ARG2" <PARAM NAME="ARG1" VALUE="value1"> <PARAM NAME="ARG2" VALUE="value2"></pre>
DMS_PRESERVE_COOKIES	[TRUE FALSE]semi-colon separated list of cookies]	<p>Set this parameter to TRUE if you want AutoVue client to pass on all cookies to the AutoVue server and integration components. Set to FALSE if you do not want AutoVue client to pass on any cookies. Specify a semi-colon separated list of cookies that the AutoVue client should pass on to the AutoVue server and integration components. If your integration relies on a cookie, it is recommended that you set DMS_PRESERVE_COOKIES to the specific cookies needed for the integration instead of setting it to TRUE.</p> <p>Default: FALSE</p> <p>Example:</p> <pre><PARAM NAME="DMSARGS" VALUE="DMS_PRESERVE_COOKIES"> <PARAM NAME="DMS_PRESERVE_COOKIES" VALUE="TRUE"></pre>

Name	Type	Value
EXTRABUNDLES	<i>name of the bundle file</i>	<p>If you are adding custom actions to AutoVue, you can specify the name of the custom resources file using this parameter. Names of the custom resource files are expected to follow: filename_XX.properties, where XX is a two-character representation of a language.</p> <p>When specifying the custom resources using this parameter, do not specify the language and the extension.</p> <p>For example: <PARAM NAME="EXTRABUNDLES" VALUE="CustomActions"></p>
EMBEDDED	[TRUE FALSE]	<p>Set to TRUE to embed the Applet in the web page. Default value: TRUE</p>
FILENAME	<i>URL</i>	Set it to the file to be opened at the applet's start-up.
For example:	upload://dir/.../file http://host/file ftp://host/file or... ftp://<user>:<password>@<ftpserver>/file server://dir/.../file	<p>Will be understood as a client local file to be uploaded on the server to be viewed.</p> <p>Specify a HTTP/HTTPS URL for file open.</p> <p>Specify a FTP/FTPS URL for file open.</p> <p>Will be understood as a server local file to be viewed. Server local files have to be located under subdirectories of the root directory specified in the jvueserver.server.directory parameter in jvueserver.properties. If the jvueserver.server.directory parameter is not set, no file will be accessible.</p>
FORMAT	[AUTO TILED METAFILE]	<p>Set the rendering format.</p> <p>TILED uses a tiled-raster representation of documents to display file.</p> <p>AUTO uses adapted representations depending on the type of file viewed.</p> <p>When set to METAFILE, data is streamed to the client as a compressed metafile (CMF). Display lists are sent to the client and the client interprets and renders these display lists.</p> <p>Generally, Office documents are rendered in TILED mode, while 2D and Raster documents are rendered in METAFILE mode.</p> <p>Default: AUTO Note: It is recommended that you do not modify this applet parameter. This parameter will be deprecated in a future release.</p>

Name	Type	Value
GUIFILE	<i>String</i>	<p>The Graphical User Interface (GUI) definition file used. Using this parameter, Web servers can customize the GUI of the applet according to client credentials. GUI files are stored in subdirectories of the root directory specified in the <code>javueservers.users.directory</code> parameter of the <code>javueserver.properties</code> file. The specification can also specify a local file using the “file://” convention.</p> <p>Note: If the GUIFILE parameter is not specified, the default AutoVue GUI is used.</p> <p>Default for the <code>javueserver.users.directory</code> parameter is <bin dir>\Profiles.</p> <p>For more information, refer to section "Customizing the GUI".</p>
HEAVYWEIGHT	[TRUE FALSE AUTO]	<p>Specify if you would like to use JOGL's heavyweight or lightweight widget to render 3D Models. When heavyweight is on, AutoVue uses hardware acceleration to render 3D. Default is AUTO and AutoVue uses heavyweight rendering on all clients except MAC and Windows 7 clients.</p>
JVUESERVER	Semicolon-separated list.	<p>Specify the list of VueServlet URLs to the AutoVue servers. Separate multiple values with a semi-colon. If multiple servers are listed, the client attempts them in a left-to-right order.</p> <p>Example: <code>http://AutoVueServer:7001/servlet/VueServlet</code></p>
LISTUSERS	[TRUE FALSE]	<p>Show list of users connected to the AutoVue server when initiating a collaboration session or when inviting users to a collaboration session.</p> <p>Default: TRUE</p>
LOCALE	[DE EN FR JA KO TW ZH]	<p>The Locale to be used in the user interface, specified as an ISO639 two-letter code. Using this parameter, Web servers can force the applet GUI to be displayed in one of the supported languages. If not set, the Locale is determined using the client system properties.</p>
LOGFILE	<i>String</i>	<p>Specify full path to the log file for messages. null is for standard output.</p> <p>Example: <code>C:\temp\clientlog.txt</code></p> <p>Default: null</p>

Name	Type	Value
ONINIT	“myFunction();”	If the ONINIT parameter function is supplied, then the AutoVue client will call the specified JavaScript function on the originating HTML page as soon as the applet has loaded and initialized. This allows for an extremely high level of control and interaction between the HTML page and the Applet. Refer to "Advanced Scripting Functionality" for more information.
ONINITERROR	“myOnErrorFunction();”	Specifies a JavaScript function to invoke if the applet fails to initialize. For example, a JavaScript function can be invoked to send an email to a system administrator when the applet initialization fails.
SESSIONXFONTPATHS	[font path]	Specify the font path to use to resolve fonts needed by the base file.
SESSIONXREFPATHS	[xref folder path]	Specify the XRefs path to use to resolve external resources needed by the base file.
SWINGLAF	String	Specify a look and feel for Swing. For example, com.java.swing.plaf.motif.MotifLookAndFeel . If null, platform’s default look and feel will be used, obtained by UIManager: getSystem LookAndFeelClassName() . Default: null Note: There are several Look and Feels available for Swing. Make sure to test the Look And Feel that you plan to use before you deploy it to production. On some Linux clients, you may need to set the LAF to the “Metal” look and feel (javax.swing.plaf.metal.MetalLookAndFeel) to have the AutoVue client working correctly.
USERNAME	String	Set it to the user name to be used for opening sessions on the AutoVue server. If not set, the applet either gets the user name from the DMS if in an integrated environment or from system properties when not integrated.
VERBOSE	OFF ERROR INFO DEBUG ALL	Set to ERROR to output all error messages. Set to INFO to display all informative messages. Set to DEBUG to display all debug messages. Set to ALL to display all messages. Set to OFF or FALSE to turn off verbosity. Default: OFF

Scripting the Applet

Basic Applet

The basic definition needed for the applet is:

```
<!-- BEGIN AutoVue Applet -->
<APPLET

<!-- NAME is optional but useful to identify the object in JavaScript -->
NAME="AutoVue"

<!-- The name of the Applet Class (not to be changed)-->
CODE="com.cimmetry.jvue.JVue"

<!-- This specifies the location of jvue.jar, jogl.jar, and gluegen-rt.jar. The WEB Browser
will download these files from this location -->
CODEBASE="http://www.webserver.com/jVue"

<!-- Name of the JAR Archive containing the Applet(not to be changed) -->
ARCHIVE="jvue.jar, jogl.jar,gluegen-rt.jar"

<!-- Optional Sizing Parameters -->
HSPACE="0" VSPACE="0" WIDTH="100%" HEIGHT="100%"

<!-- MAYSCRIPT is required. This allows the Applet to read and write a cookie identifying
sessions on the Web Browser -->MAYSCRIPT>
<!-- Set EMBEDDED to "true" for the Applet to appear within the WEB page. The default value is
"false" which causes the Applet to appear in a separate Window -->
<PARAM NAME="EMBEDDED" VALUE="false">

<!-- The VERBOSE parameter is optional. If set to "true" then diagnostic output appears in the
Browser's Java Console -->
<PARAM NAME="VERBOSE" VALUE="false">

<!-- Set FILENAME to specify the URL to load in the Applet. If not specified then the Applet
shows up with no file set initially -->
<PARAM NAME="FILENAME"VALUE="http://www.webserver.com/jVue/samples/acad12.dwg">

<!-- The JVUESERVER parameter specifies a semi-colon separated list of connection methods to
use to communicate with the the AutoVue server.Below: the client tunnels through the Servlet
installed under http://www.webserver.com/Servlet/VueServlet-->
<PARAM NAME="JVUESERVER" VALUE="http://www.webserver.com/servlet/VueServlet">

<!-- Name of the JAR Archive containing the Applet. Used by Internet Explorer -->
<!--Message for Browser that do not support Java -->
<p><b>Requires a browser that supports Java.</b></p>
</APPLET>
<!-- END AutoVue Applet -->
```

Advanced Scripting Functionality

When integrating the AutoVue applet in dynamic Web pages, all public API methods in the jVue class can be accessed through JavaScript.

For complete information on the public methods that are available, refer to the Javadocs on the AutoVue applet and the VueBean. For introductory information on the AutoVue API, refer to the *AutoVue API Introduction*.

Commonly used methods include:

Method	Description
createMobilePack(MobilePackOptions opts)	Generates the Mobile Pack according to specified options. Refer to the VueBean Javadocs for more information on this method and its options.
setFile(String url)	Set the file to be viewed in the applet.
setCompareFile(String url)	Switch to compare mode and compare the current file with a given one.
setDMSArg(String name, String value)	Set to add, modify, or remove parameters in the DMSARGS parameter list. To remove a parameter, specify 'null' value. Refer to the VueBean Javadocs for JVue.setDMSArg() for additional information.
addOverlay(String url)	Add a given file as an overlay on the current file.
printFile(PrintProperties pProps)	Print the current file using options specified. Refer to the VueBean Javadocs for information on what properties can be specified.
printFile(PrintProperties pProps, boolean UseDefaultPrinter)	Print the current file using the options specified, but do not prompt for the printer to use. Note: Control the prompting for the printer with the useDefaultPrinter parameter.
setMarkupMode(boolean enterMarkupMode)	Enter or exit Markup mode.
openMarkup(String markupID)	Open the specified Markup. If MarkupID == "*" then all Markups associated with the document are loaded. To open a local Markup specify the MarkupID as "CSI_DocName=markupName" . To open a DMAPi integrated Markup specify the MarkupID document ID as "CSI_DocID=markupID" .
collaborationInit(String sessionProperties)	Initiate collaboration session. sessionProperties - Property string describing collaboration session (has same format as applet's COLLABORATION parameter's INIT: format). See section "AutoVue Applet Parameters" for description of the Collaboration applet parameters.
collaborationJoin(String sessionProperties)	Join collaboration session in progress. sessionProperties - Property string describing collaboration session (has same format as applet's COLLABORATION parameter's JOIN: format).
collaborationEnd()	End current collaboration session.
crossProbe(String fileName)	Add a given file to the list of cross-probed files.
closeDocument()	Close current document.

Method	Description																																									
import3DFile(String fileName, HMatrix transform)	<p>Import a 3D file. Specify file name and the transformation to apply to the imported entity.</p> <p>Following is the mapping of a 4X4 transformation matrix:</p> <table><tr><td>X11</td><td>X12</td><td>X13</td><td>X14</td><td>XX</td><td>XY</td><td>XZ</td><td>X</td></tr><tr><td>X21</td><td>X22</td><td>X23</td><td>X24</td><td>YX</td><td>YY</td><td>YZ</td><td>Y</td></tr><tr><td>X31</td><td>X32</td><td>X33</td><td>X34</td><td>ZX</td><td>ZY</td><td>ZZ</td><td>Z</td></tr><tr><td>X41</td><td>X42</td><td>X43</td><td>X44</td><td>0.0</td><td>0.0</td><td>0.0</td><td>1.0</td></tr></table> <p>where (X, Y, Z) is the translation vector and the</p> <table><tr><td>XX</td><td>XY</td><td>XZ</td></tr><tr><td>YX</td><td>YY</td><td>YZ</td></tr><tr><td>ZX</td><td>ZY</td><td>ZZ</td></tr></table> <p>represents the 3X3 rotation and scaling matrix.</p> <p>The last row of the 4X4 transformation matrix should always be set to (0.0, 0.0, 0.0, 1.0).</p>	X11	X12	X13	X14	XX	XY	XZ	X	X21	X22	X23	X24	YX	YY	YZ	Y	X31	X32	X33	X34	ZX	ZY	ZZ	Z	X41	X42	X43	X44	0.0	0.0	0.0	1.0	XX	XY	XZ	YX	YY	YZ	ZX	ZY	ZZ
X11	X12	X13	X14	XX	XY	XZ	X																																			
X21	X22	X23	X24	YX	YY	YZ	Y																																			
X31	X32	X33	X34	ZX	ZY	ZZ	Z																																			
X41	X42	X43	X44	0.0	0.0	0.0	1.0																																			
XX	XY	XZ																																								
YX	YY	YZ																																								
ZX	ZY	ZZ																																								
setGUI(String guiFile)	<p>Set GUI definition file.</p> <p>Specify the name of the GUI definition file. New definition takes effect for the next file loaded in AutoVue.</p>																																									
setPage(int page)	<p>Sets the page on the currently opened document. Specify the page number to set.</p>																																									
syncMobilePack(MobilePackOptions opts)	<p>Synchronizes the Mobile Pack. The markups and intellistamp properties that were created/modified while disconnected are specified, and then checked into the DMS.</p>																																									
waitForLastMethod()	<p>Pauses current thread until last invoked method finishes execution.</p>																																									
invokeAction(String actionClassStr)	<p>Directly invokes an action (VueAction) that has no sub-actions (child actions). Refer to "Customizing the GUI" for a list of VueActions.</p>																																									
invokeAction(String actionClassStr, String subActionStr)	<p>Directly invokes a sub-action (child action) of the specified action. Refer to the default.gui file located at <AutoVue Install Root>\bin for action names.</p>																																									

Example 1:

Use the ONINIT applet parameter to automatically load a document to view, load all associated Markups and print the results.

```
<script>
<!-- Hide script from old browsers
function myFunction() {
    // The main Applet object.
    var myApp = window.document.applets["JVue"];
    // Open the specified document
    myApp.setFile('http://www.machine.com/jVue/samples/acad12.dwg');
    // Load all markups
    myApp.openMarkup('*');
    // Create a PrintProperties class
    var pPropsClass =
        myApp.getClass("com.cimmetry.common.PrintProperties");
    // Instantiate the object
    var pProps = pPropsClass.newInstance();
    // Load default properties from the user's preferences
    pProps.setProfile(myApp.getActiveVueBean().getProfile());
    // Specify the Top Center Header text: To specify a DMAPI
    //   attribute use the syntax "%X<attribute_name>"
    pProps.getHeaders().setTopCenterText("My Header");
    // Specify scaling Fit-To-Page (PrintOptions.SCALING_FIT==0)
    pProps.getOptions().setScaling(0);
    // Print the extents of the drawing (PrintOptions.AREA_EXTENTS==0)
    pProps.getOptions().setArea(0);
    // Print the document using the default printer.
    myApp.printFile(pProps, true);
    // etc...
}
-->
</script>
```

```

<!-- BEGIN AutoVue Applet -->
  <APPLET
NAME="JVue"
CODE="com.cimmetry.jvue.JVue.class"
CODEBASE="http://www.webserver.com/jVue"
ARCHIVE="jvue.jar,jogl.jar,gluegen-rt.jar"
HSPACE="0" VSPACE="0" WIDTH="100%" HEIGHT="100%"
MAYSCRIPT>
<PARAM NAME="EMBEDDED" VALUE="false">
<PARAM NAME="VERBOSE" VALUE="false">
<PARAM NAME="ONINIT" VALUE="myFunction();">
<PARAM NAME="JVUESERVER" VALUE="http://www.webserver.com/servlet/
VueServlet">
<p><b>Requires a browser that supports Java.
</b>
</p>
  </APPLET>
<!-- END AutoVue Applet -->

```

Example 2:

The frmFiles.html sample page that ships with the product makes use of the `setFile()` method to dynamically change the file in the applet.

This is easily extendible. Assuming that the HTML frame of the applet is named `AppletFrame` and that your CAD drawings are located at the URL `http://myserver/CAD`, creating four hyperlinks (HRefs) in a separate frame to dynamically call those methods will be done by adding the following lines to your HTML code:

```

<a href="JavaScript:parent.AppletFrame.JVue.setFile('http://myserver/CAD/cad.dwg')"> View
cad.dwg
</a>
<a href="JavaScript:parent.AppletFrame.JVue.setCompareFile('http://myserver/CAD/oldcad.dwg')">
Compare to old version
</a>
<a href="JavaScript:parent.AppletFrame.JVue.addOverlay('http://myserver/CAD/ovrl.dwg')"> Add
overlay ovrl.dwg
</a>
<a href="JavaScript:parent.AppletFrame.JVue.printFile(true)"> Print file
</a>

```

Configuring a Directory-Browsing Servlet for the AutoVue Client

The basic `setFile()` functionality described in section ["Customizing the Example AutoVue Client Pages"](#) allows for easy browsing of files on the server side, using the small servlet `ListDirServlet` provided with the installation. This servlet generates a list of the accessible server files in HTML format and sends it back to the client. The client can then select a file in the list and display it in the AutoVue client.

The `ListDirServlet` accepts three initialization parameters:

- `RootDir`: This is the root directory of all the directories that a user can browse on the server side.
- `RootURL`: This is the URL of the `RootDir`. Subdirectory URLs are assumed to be `RootURL + relative path to the directory`.

- **HREFFormat:** This is the format of the HRef generated for every file listed. In this format, the URL of the file listed replaces the %URL token. For example, the following default format generates a hyperlink that will trigger a setFile in the applet located in the frame named AppletFrame, for each file listed:

```
HREFFormat= JavaScript:parent.AppletFrame.setFile('%URL')
```

Because the client only receives a URL list, basic security of URL browsing still applies to the file access. However, you can also specify URLs using the pseudo-protocol 'server:' and directly browse the server file system (thus eliminating the download overhead). In order to use this protocol, you just have to ensure that the RootDir directory is also the one specified in the jvueserver.server.directory parameter of the jvueserver.properties file.

Refer to section ["AutoVue Server Configuration Options"](#) for more information.

The installation of the ListDirServlet depends on the servlet engine your Web server is using. Refer to section ["Appendix A: Deploying the VueServlet on Application Servers"](#) for more information. Once the servlet is properly installed (you can test the installation by accessing the servlet URL in a Web browser), modify the sample HTML code so that it displays the list of available files in the left frame. To do so, use the following servlet URL to access the VueServlet:

```
http://myserver/servlet/ListDirServlet
```

Note: You can set this URL as a frame in an HTML frameset.

Customizing the GUI

Choosing the GUI File

AutoVue provides you the option of customizing your graphical user interface (GUI). By default, a GUI definition file is not set and AutoVue uses an internal GUI file for the menus and toolbars. The GUI file that AutoVue generates is the same as the default.gui file located in the <AutoVue Install Root>\bin directory.

If you wish to have a customized GUI for AutoVue, you must create a custom GUI file and specify this custom file using the `GUIFILE` applet parameter. GUI files are placed at the location specified by the `jvueserver.users.directory` parameter in `jvueser.properties`. By default, the location is <AutoVue Install Root>\bin\Profiles.

Modifying the GUI

The GUI definition file describes which controls are added to which context (such as MenuBar, ToolBar, and so on).

If you are customizing your GUI file, it is recommended that you make a backup of the default.gui file and modify the controls in this file to meet your needs. The default.gui file is located in the <AutoVue Install Root>\bin folder.

If you have a previous version of AutoVue and you used a customized GUI in this previous version, we recommend that you use the diff utility to perform a comparison between the previous version's default.gui and your customized GUI. The delta between the two GUI files should be manually applied to the current version GUI.

Important: It is good practice to update your newer GUI file with the delta between the two GUI files. In order to avoid situations where some or all of the GUI elements fail to load, we recommend that you do not use the previous version's GUI file.

Structure and Syntax of GUI Files

The GUI definition file describes which controls (corresponding to available actions in the applet, like Rotate, Open, and so on) are to be added to which context (like MenuBar, ToolBar, and so on), thus allowing users to have complete control over the functionality of the applet interface.

AutoVue supports five modes: View, Compare, Markup, Collaboration, and Print Preview. A GUI file defines the graphical interface for each mode. Menu bars, toolbars, status bar and Right Mouse Button (RMB) menus are defined in this file. For some of these objects, location (north, south, west, east) may be specified. Toolbars are located in north, west or east. The status bar is always located at the bottom of the component (south).

Note: Popup menus may be added to menu bars. Menu items, popup menus or separators may be added to popup menus. Toolbars only accept buttons. Buttons or panes may be defined for the status bar. The RMB popup is processed as any other popup menu.

The following table lists each GUI keyword for each mode:

	2D	EDA	3D
View	VIEW	ECADVIEW	SMVIEW
Markup	MARKUP	ECADMARKUP	MARKUP3D
Collaboration	COLLABORATION	ECADCOLLABORATION	COLLABORATION3D
Compare	COMPARE	COMPARE	COMPARE3D

GUI Configuration Syntax

The most generic definition of a GUI file can be described through the symbols below:

- Words with CAPITAL LETTERS should be entered literally.
- The character ‘|’ is used as “or” (for example, a|b means a or b)
- The character ‘*’ means “zero or more occurrences of.”
- A GUI file can contain one or more “GUI configuration” blocks as shown in the following table:

GUI Configuration Blocks

```

GUI_configuration =
BEGIN UI VIEW UI_mode_configuration END
      {BEGIN UI COMPARE | MARKUP UI_mode_configuration END}

*UI_mode_configuration =
{menu_bar_configuration | {toolbar_configuration}* | status_bar_configuration |
RMB_popup_menu_configuration}

menu_bar_configuration =
MENUBAR BEGIN {popup_menu_configuration}* END

toolbar_configuration =
TOOLBAR NORTH|WEST|EAST BEGIN {button_control }* END

status_bar_configuration =
STATUSBAR SOUTH BEGIN {button_control | pane_control } * END

RMB_popup_menu_configuration =
RMB BEGIN {popup_menu_configuration | menu_item_control }* END

popup_menu_configuration =
POPUP IDS_{FILE|EDIT |VIEW |OPTIONS |HELP | MANIPULATE |TOOLS |ANALYSIS |MODIFY
|COLLABORATION} BEGIN {popup_menu_configuration | menu_item_control | SEPARATOR }* END

button_control =
BUTTON action_control`

menu_item_control =
MENUITEM action_control

```

GUI Configuration Blocks

```
pane_control =  
PANE action_control
```

```
action_control =  
control_name, control_key_list, permissions
```

control_name: For list of available control names refer to section ["Control Names"](#).

control_key_list: For the control key list for different controls refer to section ["Control Names"](#).

permissions: All action names need "PERM_READ".

These are the exceptions to this rule:

VueActionFilePrint needs: PERM_READ|PERM_HEADERS|PERM_WATERMARK

EcadActionSelect needs: PERM_HIDE

SMActionSelect needs: PERM_HIDE

Example:

To define a very basic user interface that only allows users, through menu items, to open or print a file and get the file information without changing watermark/headers/footers:

```
BEGIN UI VIEW  
  MENUBAR BEGIN  
    POPUP IDS_FILE BEGIN  
      MENUITEM VueActionFileOpen, , PERM_READ  
      MENUITEM VueActionFileProperties, , PERM_READ  
      MENUITEM VueActionFilePrint, , PERM_READ  
    END  
  END  
END
```

For a list of available Control Names and their functionality, refer to section ["Control Names"](#).

Control Names

The following table lists available Control Names and their functionality.

The letters in the **UI* Modes** column of the table indicate:

V - View

C - Compare

M - Markup

Control Name	UI* Modes	Functionality	Control Key List	Contexts			
				Popup Menu	Toolbar	Status Bar	RMB
VueAction FileOpen	VC	When INI option EnableUniversalFileChooser is set to 0, invokes open URL dialog. When option is set to 1, the universal file chooser dialog (that supports URLs, local files, server:// protocol and DMS files) appears. Default for EnableUniversalFileChooser is 1.		×			
VueAction FileUpload	VC	Upload local file when EnableUniversalFileChooser=0. Not available when EnableUniversalFileChooser=1.		×	×		
VueAction FileOpenUNC	VC	Open files using UNC names		×			
VueAction FileMarkup	V	Switch to Markup mode		×	×	×	×
VueAction FileCompare	V	Switch to compare mode		×			
VueAction FileOverlays	V	Launches the Overlays dialog to select and modify overlays		×			
VueAction FileProperties	VCM (M: status bar only)	Show file properties		×		×	
VueAction FilePrint	VCM	Launch the print dialog that lets you modify print options and print a file		×	×		
VueAction FileMRU	V	List most recently used documents		×			
VueAction EditSearch	VM	Launch the search dialog to perform search or repeat search		×	×		

Control Name	UI* Modes	Functionality	Control Key List	Contexts			
				Popup Menu	Toolbar	Status Bar	RMB
VueAction ViewZoom	VCM	Apply zoom	In/ Out/ Previous/ FullRes/ FitBoth/	×	×		×
VueAction ViewFlip	VCM	Apply flip	Vertical/ Horizontal/Both	×	×		
VueAction ViewRotate	VCM	Apply rotation	0/ 90/ 180/ 270	×	×		
VueAction ViewContrast	VCM	Apply contrast		×			
VueAction ViewAntiAlias	VCM	Apply anti alias		×			
VueAction ViewInvert	VCM	Apply invert		×			
VueAction ViewPage	VCM	Go to next page, previous page or select page number.		×	×		
VueAction ViewViewPoint	VC	Launches the view point dialog that lets you define a view point.		×			
VueAction ViewXrefs	VCM	Launches the XRefs dialog that lets you toggle XRefs visibility on or off.		×	×		
VueAction ViewLayers	VCM	Launches the Layers dialog that lets you toggle layer visibility on or off.		×	×		
VueAction ViewBlocks	VCM	Launches the Blocks dialog that lets you select a block to display.		×	×		
VueAction ViewViews	VCM	Launches the Views dialog that lets you select a view to display.		×	×		
VueAction ViewDrawing Info	VCM	Get the selected entity's drawing information		×			
VueAction ViewMeasure	VCM	Launches the Measurement dialog that lets you measure distance, cumulative distance, area, or calibrate		×			

Control Name	UI* Modes	Functionality	Control Key List	Contexts			
				Popup Menu	Toolbar	Status Bar	RMB
VueAction ViewSpecialViewModes	VCM	Show special view modes	Pan and Zoom Window/ MagnifyWindow/ MagnifyGlass	×	×		
VueAction ToolsDrawing Info	VCM	Get drawing information for one entity, some entities or a block Note: This feature is equivalent to the Show Entity Properties option from the AutoVue UI. For more information, refer to the <i>Oracle AutoVue User's Manual</i> .		×			
VueAction OptionsBars	VCM	Hide or show toolbars or status bar		×			
VueAction CreateMobile Pack	VM	Launches the Create Mobile Pack dialog to let you create a Mobile Pack.		×			
VueAction ReplyMobile Pack	VM	Send Mobile pack with your default e-mail client		×			
VueAction SyncMobile Pack	VM	Synchronize changes to Mobile Pack to DMS		×			
VueAction ShowRendition	VM	Show renditions in the Mobile Pack		×			
VueActionFile Browse	VM	Opens the File Browse dialog when browsing documents from different sources (local, DMS, server, URL). The document is open as soon as it is single clicked. Available only when EnableUniversalFileChooser=1 (default).			×		
VueActionFile Convert	VM	Launches the Convert dialog that lets you convert a file to different formats using convert options.			×		
VueActionFile OpenNew Window	VM	Same as VueActionFileOpen, but opens file in new window.			×		
VueAction ManageOffline Files	VM	Opens a dialog for managing offline files (adding, removing, and so on).			×		
VueAction WorkOffline	VM	Switches between offline/online modes.			×	×	

The columns indicate:

- **Control Name:** Column shows the list of available control names.
- **UI modes:** Column specifies in which modes (View, Markup, and Compare) the control names can be used.
Example: `VueActionFileOpen` can be added to View and Compare Modes, but not Markup mode.
- **Functionality:** Column specifies which functionalities are provided when this control is added to a context.
Example: Adding `VueActionFileMarkup` to any context enables you to switch to Markup mode.
- **Control key list:** Column provides the optional functionalities that can be added to a context.
 - If there is no entry for a control name in this list, it means that there is only one action to invoke. A list specifies sub-actions.
For example, for `VueActionFileOverlays`, there is no entry in the control key list, so adding it to a popup menu will provide both select and modify functionalities for overlays. The entry will look like this:

```
MENUITEM VueActionFileOverlays, , PERM_READ
```

- If there is a list of strings separated by '/', you can specify which functionalities you want added. If you don't specify any of them, by default all functionalities will be added. For example the following entry adds two buttons to the toolbar: one for Zoom In and one for Zoom Out:

```
BUTTON VueActionViewZoom, In/Out, PERM_READ
```

Whereas `BUTTON VueActionViewZoom, , PERM_READ` is interpreted as `BUTTON VueActionViewZoom, In/Out/Previous/FullRes/FitBoth, PERM_READ`

- **Contexts:** Column provides the contexts to which you can add the control to.
Example: You can have add an entry in a popup menu of the menu bar, but not in an RMB configuration

UNC File Names

When AutoVue is used in a Microsoft-based network environment, a special `VueAction()` method is available to support the viewing of files through their UNC filenames. This `VueAction()` method allows the server to directly access files on the network, as well as XRef files if they exist in the same directory as the base file. File names are specified through a specialization of the `server://` URL mechanism, where the UNC name is prefixed with the following string:

```
server://@0
```

The control name is `VueActionFileOpenUNC`. The GUI file has to be modified to use `VueActionFileOpenUNC`. For more information on the GUI file, refer to section ["Modifying the GUI"](#).

The following example shows how this `VueAction()` method works.

Example

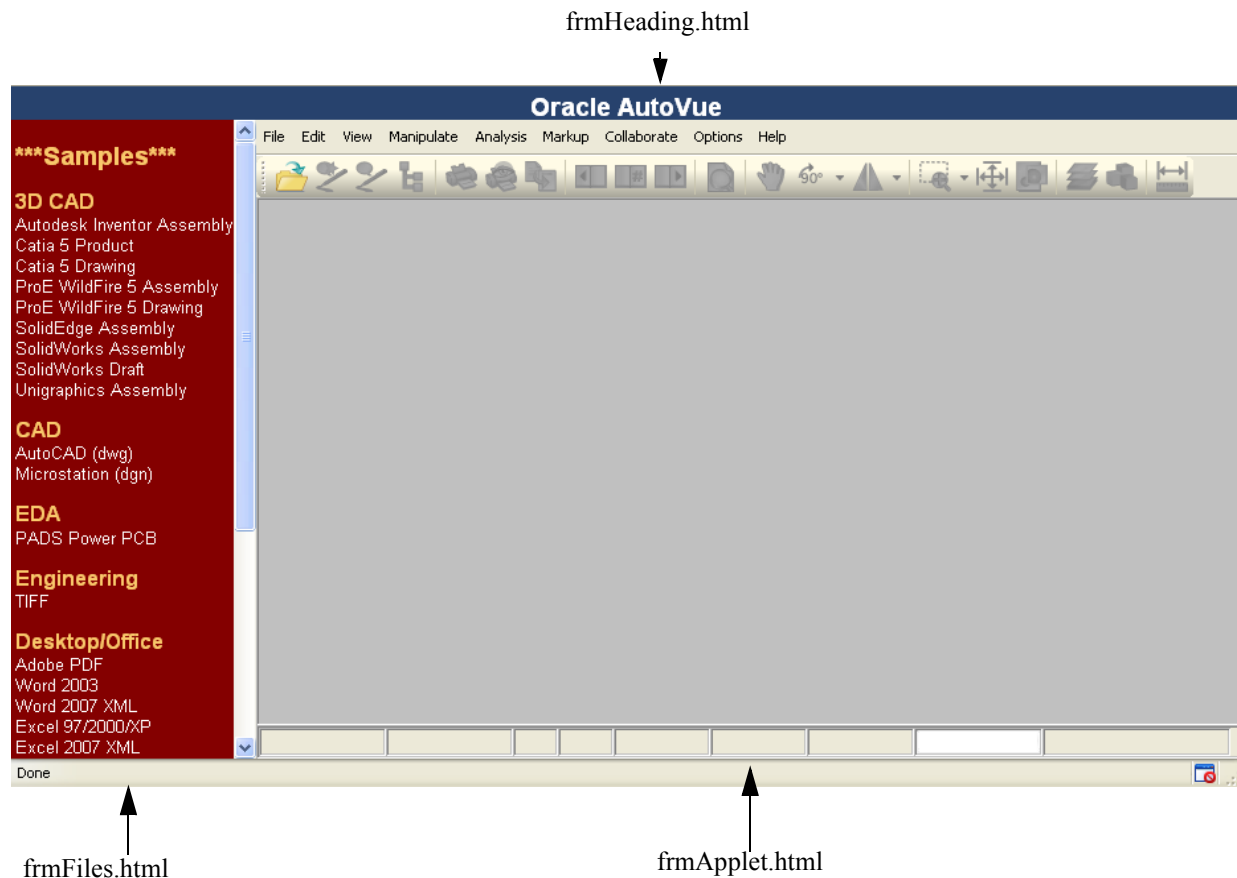
Assume that you have files on a shared network drive `\\machine1\share1`. You wish to open files that are in subdirectory `dir1` on the shared drive.

Select **Open** from the **File** menu and browse to `\\machine1\share1\dir1`. Then select a file `file1` to open. AutoVue translates this upload request to: `server://@0\\machine1\share1\dir1\file1`

Customizing the Example AutoVue Client Pages

The AutoVue client applet can be customized by setting parameters in the *frmApplet.html* file located in the <AutoVue Install Root>\html directory. The HTML code in *frmApplet.html* holds the <APPLET> tag with the customizable parameters and provides a JavaScript method called `setFile()` to allow *frmFiles.html* to dynamically change the file displayed in the applet. For more information, see section ["Scripting the Applet"](#). For a list of parameters you can set, refer to section ["AutoVue Applet Parameters"](#).

Note: During installation, if you select the Example Client Application, you can view a test HTML page *jVue.html*. The test HTML page is strictly a sample Web page and is not a required component for your deployment. This sample Web page does, however, provide a good example of how to configure the AutoVue applet. The page consists of three frames: *frmHeading.html*, *frmApplet.html* and *frmFiles.html*. The *frmFiles.html* page only appears if you choose to install the sample files during AutoVue installation.



Security Considerations for AutoVue

This section describes some security-related recommendations for AutoVue.

Installing AutoVue

We recommend that the AutoVue server is run as an unprivileged named user to ensure that direct access to the server and files on the server is restricted. Users connecting to the AutoVue server through the client can still view files and generate streaming files.

Clustered Deployments

The `DMS_PRESERVE_COOKIES` applet parameter is used when AutoVue is deployed in a clustered environment or when an integration with AutoVue relies on setting cookies and having the client pass them back as part of the request.

In AutoVue version 20.0, the `DMS_PRESERVE_COOKIES` applet parameter was updated to allow integrators to specify the exact list of cookies that the AutoVue client should pass on to the AutoVue server and/or the integration/VueLink servlet. If an AutoVue installation is being used in an integrated environment, we recommend that you make use of this enhancement to restrict the use of cookies to only those cookies that are necessary for your deployment/integration to work.

File Permissions

If configuring a `server://` directory to use with AutoVue, it is important to note that any user connecting to AutoVue has access to the files contained in the `server://` directory. Do not use the `server://` protocol if you need to restrict access to files located at the `server://` directory.

VueServlet

We recommend that the `ServerInfo` parameter of the `VueServlet` is set to `FALSE`. When set to `TRUE`, the server IP address is displayed if a user accesses the `VueServlet` page. The default value for this parameter is `FALSE`.

You can configure secure socket connection between the `VueServlet` and AutoVue server. Refer to section ["Enabling SSL Communication"](#) for information on how to set the `EnableSSL` parameter for the `VueServlet`.

Enabling SSL Communication

You can enable SSL communication between the AutoVue client and the `VueServlet` and also between the `VueServlet` and the AutoVue server.

SSL Between the AutoVue Client and the VueServlet

The `VueServlet` component is implemented as a standard Java servlet which executes within the context of an application or servlet engine. The application engine handles the communications configuration for all servlets and applications, including the provision of secure socket layer services. Secure sockets are implemented through the use of signed digital certificates and a secure handshaking procedure. Although the details of importing a digital certificate into an application server is implementation-specific, the basic process is described in the following steps.

In order to enable SSL between AutoVue client and the VueServlet, the first step is to have a valid certificate installed with your application server/web server. Then follow these steps:

- 1 Import the certificate into Internet Explorer.
- 2 Export the certificate from Internet Explorer as a base-64 encoded format and save the certificate onto the local disk. For example, into a file C:\certs.cer
- 3 Add the certificate to AutoVue's JRE using Java's keytool command:

```
C:\jdk1.6.0\bin>keytool -import -alias <servername> -file c:\certs.cer -trustcacerts -v -  
keystore C:\Oracle\AutoVue\jre\lib\security\cacerts
```
- 4 Restart the AutoVue server.
- 5 Configure the web page that embeds the AutoVue applet to point to the https:// URL for the VueServlet.

SSL between the AutoVue client and the VueServlet is now configured.

SSL Between the VueServlet and the AutoVue Server

To enable SSL between the VueServlet and the AutoVue server,

- 1 In the web.xml descriptor for the VueServlet, add the following init-param:

```
<init-param>  
<param-name>EnableSSL</param-name>  
<param-value>true</param-value>  
</init-param>
```
- 2 Make the following modification to the AutoVue server's jvueserver.properties file:

```
jvueserver.ssl.enable=true
```
- 3 Import the SSL certificate into AutoVue server's JRE and into the application server's JRE.
- 4 Restart the AutoVue server and the application server hosting the VueServlet.

SSL is now configured between VueServlet and the AutoVue server.

Identity Management Systems

An authentication facility has been added between the client and server to allow integrators to hook AutoVue to Identity Management Systems. The implementation of this facility makes use of a plug-in on the AutoVue client and another plug-in on the server. The client uses its plug-in to obtain user credentials as part of the process of connecting to the server. The client encrypts the credentials and sends them to the server which uses its plug-in to authenticate the user who is trying to connect. If the server does not recognize the credentials, it refuses the connection.

A pair of default plug-ins are now supplied with AutoVue. The *UsernamePasswordObtainer* class is supplied so that the client can prompt the user for login information (username and password). The *JAASAuthenticator* class is supplied so that the server can use the Java Authentication and Authorization Service to authenticate using the authentication mechanisms specified in the configuration text file, *jaas_authen.conf*. The default version of this file is configured to authenticate using the Kerberos protocol which is supported by Windows Active Directory and many other popular identity repository solutions.

To configure the server to use the default authentication plug-in supplied with the product, perform the following:

- 1 Edit jvueserver.properties to specify the plug-in by uncommenting the following line:

```
jvueserver.authenticator=com.cimmetry.jvueserver.JAASAuthenticator
```

- 2 Create a text file called *jaas_authen.conf* in <AutoVue Install Root>\bin directory. Add the following text in *jaas_authen.conf*:

```
/**
 * Example JAAS Login Configuration for the AutoVue server
 */
AVServer
{
    com.sun.security.auth.module.Krb5LoginModule required storeKey=true;
};
```

Edit *javueserver.properties* and add the following highlighted lines after the *-Djava.security.policy* parameter of *javueserver.cmdline*:

```
javueserver.cmdline=-Xmx128M -
Djava.security.policy="C:\Oracle\AutoVue\bin\policy"
-Djava.security.krb5.realm=<realm> -Djava.security.krb5.kdc=<kdc>
-Djava.security.auth.login.config=<full path to jaas_authen.conf>
```

Replace <realm> with your security realm.

Replace <kdc> with your key distribution center

- 3 Update the client Web pages that embed the AutoVue applet to add a new applet parameter:

```
<PARAM NAME="CREDENTIALOBTAINER"
VALUE="com.cimmetry.vueframe.UsernamePasswordObtainer">
```

- 4 Startup the AutoVue server.

- 5 Launch AutoVue client.

An authentication dialog appears and prompts you for login information. On successful login, the AutoVue client launches.

NTLM Authentication Protocol

If you are loading files that require you to enter authentication, AutoVue prompts for authentication. When using NTLM authentication, you must set the *javueserver.ntlm.enable* parameter to TRUE in *javueserver.properties*.

This change requires you to restart the AutoVue server for the change to take effect.

Integrations with AutoVue

If you are developing your own integration between the AutoVue server and a document repository (typically through the use of the ISDK that ships with AutoVue), the following points should be considered to enhance the system's overall security.

- Ensure that the original URL to a file does not contain sensitive information such as user information, server information. Use the DMSARGS applet parameter to set this sensitive information. Setting sensitive information in URLs is a potential security risk since URLs could be accessed by other users.

- Use USERNAME applet parameter to pass user information or pass user information through the CSI_UserName property that is queried by the AutoVue server at the beginning of a session. Ensure that you follow a consistent approach to passing user information.
- Always ensure that you pass valid user information to AutoVue. Handle incorrect authentication properly and enforce encryption of sensitive information.
- AutoVue version 20.0 has security enhancements. The ISDK for AutoVue 20.x leverages these security enhancements. We encourage integrators to upgrade their integrations to use the new ISDK to benefit from these security enhancements.

AutoVue Server Configuration Options

You can configure AutoVue by modifying **javueserver.properties** located in the <AutoVue Install Root>\bin directory. For example, the following sections describe options that can be configured if you wish to modify the ports that the AutoVue server is running on, or if you wish to set up a server farm or perform any other server configuration.

As of release 20.1, parameters from VueServer.ini are now set in javueserver.properties file. For information on the migration of parameters to javueserver.properties, refer to the *Release Notes*.

Note: You must restart the AutoVue server for the changes in javueserver.properties to take effect.

AutoVue Host Name Option

If you rename your server machine name after you install AutoVue, you must update this parameter:

```
javueserver.hostname = [host name]
```

Note: This new server hostname must be properly reflected in the JVUESERVER parameter specified in the VueServlet descriptors that point to this server.

RMI and Socket Ports Options

This section provides RMI and socket port parameters that may be configured. For example, the RMI port may need to be configured when setting up an AutoVue server farm, and the socket port may need to be modified to meet company policy requirements on the usage of ports within a certain range.

Parameter	Description	Default
javueserver.rmi.objectPorts = [2020-2029]	Specify a range of ports to use, or leave commented for automatic allocation.	
javueserver.socket.timeout=<integer>	Specify the inactive time in seconds after which socket times out. When 0, there is no timeout.	0
javueserver.rmi.port = <port value>	<p>The RMI port can be used to communicate with other servers when AutoVue is set up in a server farm.</p> <p>In certain situations you may need to modify the RMI port. For example, you must modify the port when the default port is used by other applications or when a company policy requires the usage of ports within a certain range.</p> <p>Note: These port numbers are not related to the HTTP port used by the Web server.</p> <p>AutoVue uses n+1 consecutive ports starting from the base RMI port, where n is the processPoolSize value specified in javueserver.properties. You should verify that the required port is open and not in use by any other process. The netstat -a program displays which ports are in use.</p>	1099

<code>javaxserver.socket.port = <port value></code>	<p>In certain situations you may need to modify the socket port. For example, you must modify the port when the default port is used by other applications or when a company policy requires the usage of ports within a certain range.</p> <p>This new socket port needs to be properly reflected in the JVUESERVER parameter specified in the VueServlet descriptors that point to this server.</p> <p>Note: This port number is not related to the HTTP port used by the Web server.</p> <p>AutoVue uses n+1 consecutive ports starting from the base socket ports, where n is the processPoolSize value specified in javaxserver.properties. You should verify that the required port is open and not in use by any other process. The netstat –a program displays which ports are in use.</p>	5099
<code>javaxserver.ssl.enable=[TRUE FALSE]</code>	<p>Specify whether to enable/disable secure socket (SSL) connections for the server. This property is required when SSL connection is enabled for the VueServlet.</p> <p>Set to FALSE to disable SSL connections for the server.</p> <p>Set to TRUE to enable SSL connections for the server.</p>	FALSE

Process Pool Size Option

The AutoVue server can run in a process pool on a single machine. The default process pool (DocServer) size is 4, and is set in the javaxserver.properties file.

Parameter	Description	Default
<code>javaxserver.processPoolSize = [integer]</code>	Set the process pool size to specify the number of DocServers to startup when the AutoVue server starts up.	4

Creating a process pool helps improve the responsiveness when handling simultaneous connections and also helps balance the load across processors in a multi-CPU machine. As a rule of thumb, you should allow for a minimum of 200MB for each process in a pool, of which approximately 50MB is for the JVM and 128MB for the Java heap. As a result, a process pool size of 4 requires at least 1GB of RAM on the machine to run comfortably. The load is balanced across the pool on the single machine.

File viewing requires memory on top of the amount for each process in the pool. Depending on the number of users and files loaded at any given time, the recommended minimum is 512MB per DocServer. Provisioning for 2GB of memory per process in the pool should be expected.

Important: Do not modify the DocServer memory settings in javaxserver.properties.

Proxy Connection Options

If the machine hosting the AutoVue server uses a proxy server to connect to the Internet, you must set the proxy setting to allow the request to go through. For example, AutoVue must connect to the Internet to retrieve required resources if missing from a file. To do so, the proxy server name must be specified in javaxserver.properties.

```
javaxserver.http.proxyhost=my.proxyserver.com:80
javaxserver.ftp.proxyhost= my.proxyserver.com:80
```

Replace `my.proxyserver.com` with the name of the proxy server running on the server and the port with the appropriate port number.

Streaming Files Options

This section provides streaming files parameters that may be configured. By setting these parameters, you can specify whether to allow streaming file generation, the maximum lifetime of streaming files, and much more.

Parameter	Description	Default
<code>juveserver.metacache.enable = [TRUE FALSE]</code>	Specifies whether to generate streaming files. When set to TRUE , streaming files are stored in the location specified by the <code>juveserver.cache.directory</code> parameter. When set to FALSE , streaming files are not generated.	TRUE
Note: The following options can be set if <code>juveserver.metacache.enable=TRUE</code> .		
<code>juveserver.metacache.pdf.enable = [TRUE FALSE]</code>	Set to FALSE : Streaming file is not generated for PDF. Set to TRUE : Streaming file is generated for PDF files. This configuration parameter should be set manually. It is recommended to set this option to FALSE as there is no benefit to enabling streaming files for PDF.	FALSE
<code>juveserver.dms.save.metafile = [TRUE FALSE]</code>	Specifies whether or not streaming files are saved in the DMS. Set to TRUE to save streaming files in DMS. Set to FALSE so that streaming files will not be saved in DMS.	TRUE
<code>juveserver.metacache.process = [TRUE FALSE]</code>	Flag for using a separate process for streaming file generation. If set to FALSE , the DocServers handle streaming file generation and the dedicated streaming file process does not start.	TRUE
<code>juveserver.metacache.threshold = [non-negative integer]</code>	Specifies the DocServer threshold at which the streaming file DocServer handles the generation of streaming files. <code>juveserver.metacache.process</code> must be TRUE for this option to take effect. Increasing this value allows the DocServer that loads a file to generate the streaming file. By default, the threshold is set to 0. That is, the streaming file DocServer generates the streaming files for all documents.	0
<code>juveserver.cache.directory=[directory path]</code>	Specifies in which directory the cached files should be saved. A central cache information file named cache.map is stored in the same directory. By default, the directory is the Cache subdirectory of the AutoVue server program directory.	<AutoVue Install Root>\bin\Cache
<code>juveserver.cache.forceascii=[0 1]</code>	Set to 1 to force the use of ASCII characters in cached files names. Set to 0 to leave characters as is. For example, you may want to use force ASCII characters if the server does not support file names with Unicode characters.	
<code>juveserver.cache.size=[value in MB]</code>	Specifies, in Megabytes, the maximum size of the file cache. The default value is 20GB. If not specified, or if value specified is less than 50 MB, a value of 4GB will be used.	20480

<code>javueserver.cache.maxlifetime = [number of days]</code>	Specifies the maximum number of days a file is kept in the AutoVue cache directory. When the maximum life time is reached, the file is deleted from the cache directory. Note: The minimum value is 1.	30
<code>javueserver.cache.maxnumfiles=[value]</code>	Specifies the maximum number of files allowed in the AutoVue cache directory. When the threshold is reached, the least recently used files are deleted. Note: The minimum value is 1000.	64000

DMS Options

This section provides DMS parameters than may be configured. However, we recommend that you do not modify these parameters.

Parameter	Description	Default
<code>dms.save.compress=[TRUE FALSE]</code>	Set to TRUE to compress save data transmitted to the DMS. Note: We recommend that you do not modify this parameter.	
<code>dms.save.length=[TRUE FALSE]</code>	Set to TRUE so that the multipart save request contain content length. Note: We recommend that you do not modify this parameter.	

Note: These options will be deprecated in the next release of AutoVue.

Collaboration Options

When using the collaboration feature in AutoVue, you can configure the following parameters.

For example: You may choose to enable the collaboration feature on the server, and/or you can specify the protocol to use for collaboration.

Parameter	Description	Default
<code>javueserver.collaboration.enable = [TRUE FALSE]</code>	Set to TRUE to enable collaboration mode on the server. Set to FALSE to disable collaboration mode. Note: If not using Real-Time Collaboration, set this parameter to FALSE.	TRUE
<code>javueserver.collaboration.protocol=[RMI JXTA]</code>	Specifies the protocol connection between AutoVue servers in a server cluster. Note: RMI is the default. JXTA can be used if there is a firewall between the servers.	RMI
<code>javueserver.collaboration.tcp.port=[integer]</code>	BaseTCP port to be used. Note: The configuration parameters below need to be changed when using more than one server cluster in a server farm.	9700

Parameter	Description	Default
<code>javueserver.collaboration.id.min=[integer]</code>	Minimum ID given to users and collaboration sessions by this server. Change this ID when you are running many AutoVue servers that must communicate together for collaboration. The second server must have a minimum ID of at least <code>javueserver.collaboration.id.min + javueserver.collaboration.id.range</code> of the first server. Otherwise, an ID overlap may occur.	0
<code>javueserver.collaboration.id.range=[integer]</code>	Range of IDs given to users and collaboration sessions by this server. This will limit the number of simultaneous connections.	100000
<code>javueserver.collaboration.group=[group name]</code>	Specify the group name for servers when using JXTA connection for collaboration.	

Offline Mode Option

This section provides offline mode parameters that may be configured. For more information on offline mode, refer to section ["AutoVue in Online/Offline Mode"](#).

Parameter	Description	Default
<code>javueserver.update.xml.url=[url to autovueupdate.xml]</code>	Specify the URL to the <code>autovueupdate.xml</code> file that is required for the offline mode installer.	

NTLM Authentication Option

To support NTLM authentication, set the following parameter in `javueserver.properties` to `TRUE`:

```
javueserver.ntlm.enable = [TRUE | FALSE]
```

Note: The default value is `FALSE`.

log4j and Diagnostics Options

This section provides log4j and diagnostics parameters that may be configured. These parameters can be set to configure the logging level and time interval for detecting log4j configuration changes, and the output diagnostics information.

Parameter	Description	Default
<code>javueserver.log4j.configureandwatch = [TRUE FALSE]</code>	Set this to <code>TRUE</code> to be able to dynamically change the log4j logging level.	<code>FALSE</code>
<code>javueserver.log4j.configureandwatch.delay=[integer]</code>	Time interval for waking up and detecting log4j configuration change.	60
<code>javueserver.diagnostics.format=[xml text]</code>	Specify the output format for the AutoVue server diagnostics.	xml

<code>javueserver.diagnostics.period=[interval in minutes]</code>	Specify the interval in minutes at which the AutoVue server diagnostics are generated.	No default; diagnostics are generated on demand.
---	--	--

Reboot Option

The following table describes the reboot parameter for DocServers in `javueserver.properties`. By setting this parameter, you can control the reboot time interval for DocServers.

Parameter	Description	Default
<code>javueserver.reboot.timeout=<interval in minutes></code>	If a DocServer is idle for the time specified by this parameter, the DocServer is rebooted. The default time out is 30 minutes.	30

Recovery Attempt Option

The following table describes the recovery attempt parameter in `javueserver.properties`. By setting this parameter, you can control the number of recovery attempts for the DocServer.

Parameter	Description	Default
<code>javueserver.recovery.attempts=[integer]</code>	Specify the number of recovery attempts for the DocServer when an exception is thrown. After this number of failed recovery attempts, the DocServer restarts.	5

DLL Version Option

You can specify user-defined DLLs with the following parameter.

Parameter	Description	Default
<code>javueserver.version.extralibraries=DLL_1;DLL_2;DLL_3;...</code>	Specify a semi-colon separated list of user-defined DLLs. AutoVue will list the versions of these DLLs in the Help > About dialog.	

File Format Information Option

On startup, AutoVue registers all of its components into a `VueServer.ini` file. You can specify an alternate path for `VueServer.ini` using this option.

Parameter	Description	Default
<code>javueserver.inifile=[file name]</code>	Specify the INI file where AutoVue stores information on the file formats supported by AutoVue. By default, it is <code>VueServer.ini</code> located in the <AutoVue Install Root>\bin directory. Note: AutoVue saves certain memory management settings in this file. We recommend that you do not modify these options or the file.	<code>VueServer.ini</code>

Global User Options

The following global user settings may be configured. These parameters specify the directory in which user information is stored, and the names for global configuration files.

Option	Description	Default
javueserver.users.directory	Contains the directory in which user information is stored (initialization files and GUI files).	<AutoVue Install Root>\bin\Profiles
javueserver.users.defaultini	AutoVue provides a way to push certain INI settings to the user INI the first time the user accesses AutoVue. This is done by setting the required options in the default.ini file or in the file specified by autovue.users.defaultini parameter. This file should be located at <AutoVue Install Root>\bin directory.	default.ini
javueserver.users.allusersini	AutoVue provides a way to push INI settings to the user profile every time a user accesses AutoVue. This is done by setting required options in allusers.ini (or the files specified by autovue.users.allusersini). This file should be at <AutoVue Install Root>\bin\Profiles directory.	allusers.ini
javueserver.users.timeout=[interval in seconds]	Specify the user session timeout in seconds. If the user session is idle for the specified time period, the session is closed.	1800

Markup Options

You can configure the Markup Files dialog, Markup Files directory, permissions, and markup symbols library by setting the following options.

Option	Description	Default
javueserver.markup.nativegui.type	Add Author, Date, and Markup Info columns to the Markup Files dialog. 0: Name column displays 1: Enable Author 2: Enable Date 4: Enable Markup Info Note: These are <i>ORed</i> flags. For example: Enter 7 to enable all three columns.	0
javueserver.markups.directory	Specifies in which directory the Markup files should be saved. Markups are saved with random names in this directory, and the mapping between Markup files and their base file is held in a central map file named markups.map , stored in the same directory. Note that multiple servers should not share the same location for storing markups. Note: This option is for server-managed markups.	<AutoVue Install Root>\bin\Markups

Option	Description	Default
javueserver.markups.permissions	By default, all users can see the Markups of a file but only the owner of a Markup can modify it. The Permissions key can be used to change that behavior: setting it to 0 allows all users to see and change Markup files.	
javueserver.markups.symboldir	Specifies in which directory the Markups symbol libraries are stored. By default, the directory is the symbols subdirectory of the AutoVue server program directory.	<AutoVue Install Root>\bin\Symbols

Server Viewable Local Files Options

You can configure the directories in which to search for local files by setting the following options.

Option	Description	Default
javueserver.server.directory=[directory path]	Specifies a directory accessible to the server that clients can access. This key has to be set to allow the client to see the server local files through the 'server://' pseudo-protocol. Refer to the FILENAME description in "AutoVue Applet Parameters" . By default no server files can be viewed. Setting this key allows users to see ANY local file in the specified directory and subdirectories. However, the server takes care of parent references in paths (the ".." directory) to avoid security breaches.	
javueserver.server.directory[n]=[directory path]	To specify multiple directories, specify Directoryn=[Path]. To access files at these locations, specify "server://@n/..."	
javueserver.server.browse.enable=[TRUE FALSE]	Specifies whether to allow users to browse the server directories. Set to TRUE to allow users to browse the server directories.	TRUE

Online Help Options

You can specify the entry points for language-specific online help by setting the following options.

Option	Description
javueserver.help.file_en	Entry specifies the URL to the English Help file. If Online Help does not exist for a language, AutoVue loads the English help file by default.
javueserver.help.file_xx	Entry specifies the URL to the Help file for the language "xx."

Memory Optimization

AutoVue performs memory management when loading large files. If AutoVue memory hits a pre-defined threshold, AutoVue dumps the least-used data from memory to the disk. This memory management scheme helps load larger models in AutoVue 20.1. Memory management is enabled by default. To disable it, you must set `javueserver.memory.managed=FALSE` in `javueserver.properties`. Refer to the following table for all memory management-related configurations.

In `javueserver.properties`, you can set the following parameters to optimize memory or performance speed.

Parameter	Description	Default
<code>javueserver.memory.managed = [TRUE FALSE]</code>	<p>This option orients the optimization in the product towards speed or memory.</p> <p>If set to FALSE, speed is optimized.</p> <p>If set to TRUE, memory is optimized. Setting this option to TRUE does not impact the loading of Office and Raster formats.</p> <p>Note: When this option is set to TRUE, AutoVue's memory manager dumps the least used components from memory onto disk when the process memory hits the threshold specified in <code>javueserver.memory.threshold</code>.</p> <p>Effect on Performance: Performance speed is improved if value is set to FALSE. If loading large files, or files that require a lot of memory, we recommend that you set the option to TRUE to optimize memory usage.</p>	TRUE (for Windows OS) FALSE (for Linux OS)
<code>javueserver.memory.threshold= value</code>	<p>Specifies the process memory threshold for AutoVue after which the memory manager dumps data. Specify value in MB.</p> <p>When set to 0, AutoVue calculates the memory threshold based on the following formula:</p> <p>$[(\text{Total memory on the machine})/(\text{n}+1 \text{ where n is processpoolsize})]*1.2$</p> <p>The computed value does not exceed 1GB or the maximum memory size addressable for the process on the system multiplied by 0.8, whichever is less.</p> <p>Note: Minimum value is 256MB</p> <p>Effect on Performance: Performance speed is improved when the threshold is a larger value. To optimize memory usage, set a lower threshold.</p>	0

AutoVue programmatically writes certain memory management options in VueServer.ini. These options should not be modified:

Parameter	Description	Default
MNGMEMPPAGESIZE=[num]	When memory management is enabled, specify the size of pages (memory) to allocate when storing the managed data. Each memory page is predefined. num = number of bytes used to allocate pages in memory. Minimum value: 8192 (8KB) Maximum value: 1048576 (1MB) Note: The memory pages are dumped to temporary dumping files located in the path defined in MNGTEMPFNAME.	131072 (128KB)
MNGTEMPFNAME=[folder location]	When memory management is enabled, specifies the location and name of the temporary dumping folder. If the temporary dumping folder does not exist, the folder is created and marked for deletion.	<AutoVue Install Root>\bin\avd ump

Linux-Specific Options

The following section list Linux-specific parameters that can be configured in jvueserver.properties.

Preload Java Class Option

The following table describes java class preload parameter in jvueserver.properties.

Parameter	Description	Default
jvueserver.preload =[preloader class name]	Enables loading of specified java class prior to the AutoVue server startup.	com.cimmetry. jvueserver.util. UnixPreloader

Xvfb Options

AutoVue provides the following options to initialize Xvfb parameters:

- xvfb.display
- xvfb.process
- xvfb.policy
- xvfb.colormap
- xvfb.args

It is recommended that you do not modify these options. However, in the event of a port conflict, modifying xvfb.display may resolve the issue. Refer the following table for more information on xvfb.display.

Option	Description	Default
xvfb.display=<port number>	Specifies the initial port to use for Xvfb.	909

WINE Options

AutoVue provides the following options to configure WINE parameters:

- wine.dir
- win.config.dir
- win.config.file

It is recommended that you do not modify these options.

OEM Copyright Notice

AutoVue provides an option to replace the default copyright in the Help dialog with a custom copyright:

Parameter	Description	Default
oem.copyright.notice=	Specify a notice to replace the default Oracle copyright in the Help About dialog.	

VueServlet Configuration Options

The following table describes VueServlet initialization parameters that can be set in web.xml. For more information, refer to section ["VueServlet Deployment Instructions"](#).

Parameter	Description	Default
DebugLevel=[0-100]	Set the debug output category.	0
EnableSSL=[TRUE FALSE]	Set to TRUE to enable secure socket connection to the AutoVue servers.	FALSE
EnableEM=[TRUE FALSE]	Specify whether or no to retrieve Oracle Enterprise Management information. Set to TRUE to retrieve information. Set to FALSE to disable information retrieving.	FALSE
JVueServer=[server host names]	A semicolon separated list of the AutoVue server host names. This parameter is used by the VueServlet to connect to the AutoVue servers through a socket connection. The JVUESERVER parameter needs to be set to the hostname:port value used when starting the AutoVue Server. This port value must match the port set in jvueserver.properties. You can specify more than one hostname:port separated by semi-colons (;) for fail-over. In other words, if one machine is down the servlet will try the next machine. If JVUESERVER is not specified, it defaults to localhost:5099. The servlet assumes that the AutoVue server is running on the same machine as the Web server and communicates through port 5099. Note: The port listed in this option should match the port listed in the jvueserver.socket.port option in the jvueserver.properties file.	local host name:5099
InvokerCount=[value]	Set the number of simultaneous connections from the VueServlet to the AutoVue server. If the number of pending requests at any given time exceeds this number, then the remaining requests wait in a queue until a connection is free.	100
ServerInfo=[TRUE FALSE]	Set to TRUE to include the AutoVue server information on VueServlet status page. Set to FALSE to hide the AutoVue server information.	FALSE
Verbose=[TRUE FALSE]	Set to TRUE to enable debug output to the application server log. Set to FALSE to disable debug output.	FALSE

Appendix A: Deploying the VueServlet on Application Servers

The VueServlet allows the AutoVue client to communicate with the AutoVue server using HTTP tunneling. This has two advantages:

- The client and the AutoVue server can generally communicate across firewalls since the standard HTTP ports (for example, 80) are used.
- The client can be configured to use the HTTPS protocol to communicate with the VueServlet. This ensures that all communications are secure.

the AutoVue client encodes requests from the HTTP/HTTPS protocol and attempts to invoke the VueServlet on the specified server. The VueServlet decodes the parameters included in the request and forwards the request to the AutoVue server using a socket connection. The VueServlet also replies to the client machine using the same HTTP/HTTPS protocol. You can deploy the VueServlet with any application server you choose. For a list of application servers that are certified by Oracle, refer to section ["System Requirements"](#).

The exact steps to set up the VueServlet on your application server depend on the software you are using. This section describes the steps to setup the VueServlet for several popular Application Servers/Servlet Engines. Generally, you can follow similar steps to deploy with any application server. Refer to your application server documentation for specific instructions.

For information on configuring the VueServlet, refer to section ["VueServlet Configuration Options"](#).

Generic Steps to Deploy the WAR File

- 1 Launch the administrative console of your application server.
- 2 Select **Install a new Web application**.
- 3 Browse and select `VueServlet.war`.
- 4 Specify `VueServlet` for the context name.
- 5 Deploy **VueServlet.war**.

We provide you with instructions for deploying **VueServlet.war** with some application servers in the following section.

Deploying the WAR File with WebLogic 9.x and up

- 1 Logon to the Administrative Console for WebLogic.
- 2 Select **Deployments** from the tree.
- 3 Click on **Install**.
- 4 Browse to the folder containing `VueServlet.war` and select **VueServlet.war**.
- 5 Enter `VueServlet` for the **Application Name**.
- 6 Select the Server to which you wish to deploy `VueServlet`.
Example: `myserver`
- 7 Click **Activate Changes**.
- 8 Select **Deployments** again and select the `VueServlet` application.
- 9 Click **Start** and select **Servicing all requests**.

The application starts.

Once the deployment is successful, verify the deployment. To do so, connect to:

`http://<host name>:<port>/VueServlet/servlet/VueServlet`

where `<host name>` is the name of your Application Server host machine and `<port>` is the port your application server is running on.

Deploying the VueServlet with Tomcat 6.x and up

- 1 Copy `vueservlet.war` to your Tomcat *webapps* directory.
- 2 Restart Tomcat.

The VueServlet is deployed automatically.

Deploying the WAR File with WebSphere 6.1 and up

- 1 Launch the administrative console and log on to the application server.
- 2 Select **Applications** and then **Install new application**.
- 3 Browse and select `VueServlet.war`.
- 4 Specify **VueServlet** for the context name and click **Next**.
- 5 Accept the default values in the screen that appears.
- 6 In the **Install New Application** screen, enter **VueServlet** for the **Application Name** and click **Next**.
- 7 Accept the default values in the remaining screens. Then click **Finish**.
- 8 To start the VueServlet application, go to **Applications** and then **Enterprise Applications**.
- 9 Select **VueServlet** and click **Start**.

To test the VueServlet, connect to:

`http://<host name>:<port>/VueServlet/servlet/VueServlet`

where `<host name>` is the name of your application server host machine and `<port>` is the port your application server is running on.

Deploying the VueServlet on non-J2EE Application Servers

Setting up the VueServlet

Below are generic instructions for deploying the VueServlet with a non-J2EE application server.

For Tomcat users, refer to section ["Deploying the VueServlet with Tomcat 6.x and up"](#).

- 1 Copy the file `vueservlet.jar` to your Servlet Engine's `servlet` directory.
- 2 Add `vueservlet.jar` to your Servlet Engine's `CLASSPATH`.
- 3 Create an alias for VueServlet to `com.cimmetry.servlet.VueServlet`.
- 4 If your AutoVue server is running on a different machine, specify the init parameter `JVueServer` to be `my.jvueserver.com:5099` where `my.jvueserver.com` specifies the machine on which AutoVue server is running. 5099 specifies the socket port that the AutoVue server uses. If the server is using a different socket port, specify the correct socket port in parameter `JVueServer`.

- 5
- 6 For the changes to take effect, restart the servlet engine.
Note: The default socket port is 5099.

Deploying on Jetty

- 1 Add `VueServlet.jar` to Jetty's class path.
- 2 Edit `startjetty.bat` and add the full path to `VueServlet.jar` to the `CLASSPATH` variable.
- 3 Edit `webdefault.xml` and add the following:

```
<servlet id="VueServlet">
  <servlet-name>VueServlet
</servlet-name>
  <servlet-class>com.cimmetry.servlet.VueServlet
</servlet-class>
  <init-param>
    <param-name>JVueServer
    </param-name>
    <param-value>www.jvueserver.com:5099
    </param-value>
  </init-param>
  <init-param>
    <param-name>Verbose
    </param-name>
    <param-value>>false
    </param-value>
  </init-param>
  <init-param>
    <param-name>DebugLevel
    </param-name>
    <param-value>0
    </param-value>
  </init-param>
  <load-on-startup>0
</load-on-startup>
</servlet>
```

- 4 Replace `www.jvueserver.com` with the name of the machine on which the AutoVue server is running. 5099 specifies the socket port that the AutoVue server uses. If the server is using a different socket port, specify the correct socket port.
- 5 Start Jetty and the AutoVue server.
- 6 Test that the `VueServlet` is installed properly; Open a Web browser and enter the URL to the `VueServlet`:
`http://<machine name>:5098/servlet/VueServlet`

Configuring the ISAPI Redirector on Windows IIS

Windows IIS normally cannot execute servlets. Configuring the IIS to use ISAPI redirector allows the IIS to send servlet requests to the application server. Follow the instructions for the application/servlet engine to setup ISAPI redirection.

Appendix B: Non-Interactive Installations

Installation

To install AutoVue in non-interactive mode, you need to specify a configuration file that contains the required installation parameters. To do so, you must generate the configuration file manually with the following syntax:

For Windows OSes:

```
#Specify Installation Directory
#-----
USER_INSTALL_DIR=C:\\Oracle\\AutoVue

#Select Shortcut Folder
#-----
USER_SHORTCUTS=C:\\Documents and Settings\\Administrator\\Start
Menu\\Programs\\Oracle AutoVue

#Select Features (Available: ProgFiles, AdminDocs, UserDocs, Website, SampleFiles, APIEx)
#-----
CHOSEN_INSTALL_FEATURE_LIST=ProgFiles, AdminDocs, UserDocs, Website, SampleFiles, APIEx

#Specify host name for AutoVue Server
#-----
JVUESERVER_HOST=avserver1

#Specify settings for Web Server
#-----
WEBSERVER_HOST=avserver1:80
WEBSERVER_DOCROOT=C:\\inetpub\\wwwroot
WEBSERVER_PATH=AutoVue
```

For Linux OSes:

```
#Specify Installation Directory
#-----
USER_INSTALL_DIR=/home/apps/AutoVue

#Select Features (Available: ProgFiles, AdminDocs, UserDocs, Website, SampleFiles, APIEx)
#-----
CHOSEN_INSTALL_FEATURE_LIST=ProgFiles, AdminDocs, UserDocs, Website, SampleFiles, APIEx

#Specify host name for AutoVue Server
#-----
JVUESERVER_HOST=avserver

#Specify settings for Web Server
#-----
WEBSERVER_HOST=avserver
WEBSERVER_DOCROOT=/var/www/html
WEBSERVER_PATH=AVclient
```

The following are installation parameters that you can specify in the configuration file:

Parameter	Description	Default Value
USER_INSTALL_DIR=[file path]	Specify the path where you want to install the AutoVue server.	
USER_SHORTCUTS=[file path]	Specify the shortcut path. Note: This parameter is only for Windows OS installations.	
CHOSEN_INSTALL_FEATURES=[ProgFile, AdminDocs, UserDocs, Website, SampleFiles, API Ex]	Specify the features to install. The comma-separated list can contain the following features: ProgFile: Installs Oracle AutoVue. AdminDocs Installs Oracle AutoVue system administration documentation. UserDocs: Installs AutoVue end-user documentation. Website: Installs AutoVue client components onto a Web server. SampleFiles: Installs sample files. APIEx: Installs examples of how Oracle AutoVue features can be added to third-party applications using APIs.	ProgFile, AdminDocs, UserDocs, W ebsite
JVUESERVER_HOST=[AutoVue Server host name]	Specify the AutoVue server host name.	
WEBSERVER_HOST=[Web server host name and port]	Specify the Web server host name.	
WEBSERVER_DOCROOT=[Web server document root]	Specify the document root of the Web Server	
WEBSERVER_PATH=[Path to AutoVue client files relative to DOCROOT]	Specify the path to the AutoVue client files relative to the document root.	

After you specify the parameters for the configuration file, you can run the installation in non-interactive mode. Enter the following command lines:

For Windows OSes:

```
InstallClientServer.exe -i silent -f <full path to configuration file>
```

For Linux OSes:

```
InstallClientServer_lin.bin -i silent -f <full path to configuration file>
```

Uninstallation

If AutoVue is installed in non-interactive mode, the uninstallation is automatically in non-interactive mode. Simply invoke the uninstaller for AutoVue:

For Windows OSes:

```
<AutoVue Install Root>\uninstall\uninstall.exe
```

For Linux OSes:

```
<AutoVue Install Root>/uninstall/uninstall
```

Appendix C: Configuring AutoVue Plug-in for Enterprise Manager

An AutoVue plug-in can be added to Oracle Enterprise Manager to enable monitoring of AutoVue servers.


Prerequisites

- Oracle AutoVue 20.1 Client/Server Deployment
- Oracle Enterprise Manager 11g

Important: The plug-in is configured to work with Jetty that is included with AutoVue server. If you plan to use the plug-in to monitor AutoVue server usage, make sure to startup Jetty that is included with AutoVue. You must set the `VueServlet` parameter `EnableEM` to `TRUE` in Jetty's `webdefault.xml`.

Since the plug-in reports sensitive information about the AutoVue server, it is recommended that the `VueServlet` instance that communicates with Enterprise Manager is secure and is within the intranet. This `VueServlet` instance should only be used for Enterprise Manager reporting.

Installing the Plug-in

- 1 Connect to Oracle Enterprise Manager for a Web browser.
- 2 Enter user login information.
- 3 From the Oracle Enterprise Manager Grid Control 11g home page, click **Setup**.
The Overview of Setup page appears.
- 4 From the Overview of Setup section, click **Management Plug-ins**.
The Management Plug-ins page appears.
- 5 Click **Import**.
The Import Management Plug-ins page appears.
- 6 To import the plug-in, click **Browse**.
- 7 Select the plug-in, `oracle_autovue.jar`, from located in the `bin` directory and then click **OK**.
- 8 Click the Deploy icon  .
- 9 Click **Add Agents**.
The Search and Select: Agents page appears.
- 10 Click **Go**, select an agent from the results, and then click **Select**.
The Deploy Management Plug-in: Select Targets page appears.
- 11 Click **Next**.
The Deploy Management Plug-in: Review page appears.
- 12 Click **Finish**.
- 13 From the Oracle Enterprise Manager Grid Control 11g home page, click **Setup**.
The Overview of Setup page appears.
- 14 Click **Agent**.
The Management Agents page appears.
- 15 Click the agent link **hostname:3872**.

- 16 From the Add drop-list, select **Oracle AutoVue** and then click **Go**.
The Add Oracle AutoVue page appears.
- 17 Enter the following information:
 - A descriptive name in the Name field (for example, *AutoVue server*).
 - The name or the IP address of the machine where the VueServlet that enables EM is installed.
 - The servlet port number (for example, *:5098*) in the Oracle AutoVue Server Servlet Port field.
- 18 Click **OK**.
- 19 Edit Jetty's webdefault.xml and VueServlet parameter EnableEM and set it to TRUE (lines in bold below):

```
<servlet id="VueServlet">
  <servlet-name>VueServlet</servlet-name>
  <servlet-class>com.cimmetry.servlet.VueServlet</servlet-class>
  <init-param>
    <param-name>JVueServer</param-name>
    <param-value>mtloaqal3.cimmetrysystems.com:5099</param-value>
  </init-param>
  <init-param>
    <param-name>Verbose</param-name>
    <param-value>>false</param-value>
  </init-param>
  <init-param>
    <param-name>EnableEM</param-name>
    <param-value>TRUE</param-value>
  </init-param>
  <init-param>
    <param-name>DebugLevel</param-name>
    <param-value>0</param-value>
  </init-param>
  <load-on-startup>0</load-on-startup>
</servlet>
```

- 20 Restart Jetty for the changes to take effect.
- 21 From Enterprise Manager's Monitored Targets section, click the created AutoVue target (for example, *AutoVue server*) to view AutoVue server status information.

Note: If EnableEM on the VueServlet is set to FALSE, the Oracle Enterprise Manager will not be able to retrieve the AutoVue server status and will keep pinging the server through the VueServlet. In this situation, you will see the following message on the VueServlet: *Enterprise Manager is not enabled*.

Appendix D: Samples and API Examples Included with AutoVue

During the installation process, if you select **Custom** installation, AutoVue provides you options to install samples and API examples. This chapter provides an overview of the samples and API that are installed with AutoVue.

API Examples

The following API Examples are installed at `<AutoVue Install Root>\examples` if you do a Custom installation and choose to install API examples. To test these samples make sure AutoVue server and the application server hosting the VueServlet are running. You will need to update the samples with the correct URL to the VueServlet. By default, the samples use Jetty installed with AutoVue:

- **AWTSample**
This example demonstrates how to build a basic AutoVue application using the VueBean API.
- **VueActionSample**
This provides an example of how to implement hotspots using the VueAction. For more information on VueAction, refer to the *AutoVue API Programmer's Guide* and to the *VueBean JavaDocs*.
To use this sample:
 - Hotspots.txt contains some hotspot definitions. You can use these definitions to test this sample or you can create your own definitions.
 - PartCatalogueAction.java and PartListAction.java demonstrate how to write custom actions.
 - A custom gui file is the customized gui for the custom actions. Copy this file to `<AutoVue Installation Directory>\bin\Profiles`.
 - Update the Java code as needed and compile the code
 - Bundle all the class files into VueActionSample.jar
 - Run this sample using the following command:

```
java -cp <full path to jvue.jar>;<full path to VueActionSample.jar>
com.cimmetry.jvue.JVue -param GUIFILE=<path>/custom.gui -param EXTRABUNDLES=/
PartCatalogueAction
```

Note: The *Hotspots API Guide* provides information on how to implement AutoVue's hotspots API using JavaScript.

Sample Files

When you choose to install sample files during the AutoVue installation process, the following samples are installed:

- Sample 2D, 3D, EDA, Office and Graphics file are installed at `<AutoVue Install Root>\html\samples`
- Sample web pages to invoke the AutoVue applet are installed at `<AutoVue Install Root>\html`. You will need to update the JVUESERVER and the CODEBASE parameters to the URL to the VueServlet and the location of the client JAR files, respectively. You will need to update files frmApplet.html and frmFiles.html. The web page to invoke from the client is jVue.html.

Note: If a web server is detected on the machine where you install AutoVue, the samples and the web pages are copied to the web server doc root.

- A Batch Printing JavaScript is also installed at `<AutoVue Install Root>\html`. The files is batchPrint.html. Make sure to update the JVUESERVER and the CODEBASE parameters in order to use this sample.
- frmFiles.html contains the following JavaScript samples:
 - 2D Comparison
 - 3D Comparison
 - Overlay

- MockUp
- Cross-Probe
- Printing

Feedback

If you have any questions or require support for AutoVue please contact your system administrator.

If at any time you have questions or concerns regarding AutoVue, call or e-mail us.

General Inquiries

Telephone: +1.514.905.8400 or +1.800.363.5805

E-mail: autovuesales_ww@oracle.com

Web Site: <http://www.oracle.com/us/products/applications/autovue/index.html>

Sales Inquiries

Telephone: +1.514.905.8400 or +1.800.363.5805

E-mail: autovuesales_ww@oracle.com

Customer Support

Web Site: <http://www.oracle.com/support/index.html>