Java Dynamic Management Kit 4.2 Installation Guide and Release Notes

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303-4900
U.S.A.

Part Number 806-6632–10
December 2000

Contents

Preface

The Java Dynamic Management™ Kit provides a set of Java™ classes and tools for developing management solutions. This product conforms to the Java Management extensions (JMX™), v1.0 Final Release, which defines a three-level architecture: resource instrumentation, dynamic agents and remote management applications. The JMX architecture is applicable to network management, remote system maintenance, application provisioning, and the new management needs of the service-based network.

This installation guide will help you to install and run the Java Dynamic Management Kit 4.2 on your machine. This product is supported on the Solaris™ operating environment and Windows NT operating environment.

This document contains:

- Instructions for Installation on a Solaris Platform.
- Instructions for Installation on a Windows NT Platform.
- Release Notes for all platforms.

Getting Help With Your Installation

If you have problems with any part of the product installation process, call your authorized service provider. Have the following information ready:

- The model number of your machine.
- The serial number of your machine.
- The Windows NT or Sun Solaris operating system release number.
- The product release number: 4.2

- The product serial number shown on the label or the serial number you were given when downloading the software.

On the Solaris platform, use the uname command to display the SunOS operating system release number:

```
prompt% uname -sr
SunOS 8
```

On the Windows NT platform, click the Help menu of any Windows NT Explorer window and select the About Windows NT item.

Ordering Sun Documents

Fatbrain.com, an Internet professional bookstore, stocks select product documentation from Sun Microsystems, Inc.

For a list of documents and how to order them, visit the Sun Documentation Center on Fatbrain.com at http://www1.fatbrain.com/documentation/sun.

Accessing Sun Documentation Online

The docs.sun.com^{SM} Web site enables you to access Sun technical documentation online. You can browse the docs.sun.com archive or search for a specific book title or subject. The URL is http://docs.sun.com.

Typographic Conventions

The following table describes the typographic changes used in this book.

TABLE P–1 Typographic Conventions

Typeface or Symbol	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls –a` to list all files. `machine_name% you have mail.`

`AaBbCc123`	What you type, contrasted with on-screen computer output	`machine_name%` **`su`**
		`Password:`

AaBbCc123	Command-line placeholder: replace with a real name or value	To delete a file, type **`rm`** *filename.*
AaBbCc123	Book titles, new words, or terms, or words to be emphasized.	Read Chapter 6 in *User's Guide.* These are called *class* options. You must be *root* to do this.

Shell Prompts

The following table shows the default system prompts on the supported platforms. Examples on the Solaris platform use the Korn shell unless otherwise noted.

TABLE P–2 Shell Prompts

Shell	Prompt
C shell prompt	`machine_name%`
C shell superuser prompt	`machine_name#`
Bourne shell and Korn shell prompt	`$`
Bourne shell and Korn shell superuser prompt	`#`
Windows NT system prompt	`C:\>`

CHAPTER 1

Installation on a Solaris™ Platform

The steps for installing and starting the Java Dynamic Management Kit in a Solaris™ environment are described in these sections:

- "Planning Your Installation" on page 11
- "Installing the Software on a Solaris Platform" on page 13
- "Setting Environment Variables in a Solaris Environment" on page 18

Planning Your Installation

Plan your installation to determine where to install the packages supplied with the product and to ensure that your machine meets the installation requirements.

Upgrading from Previous Versions

If you are upgrading your Java Dynamic Management Kit from versions 3.0, 3.2, 4.0, or 4.1, you are not required to uninstall the earlier version. Version 4.2 has a different installation structure that allows both old and new versions of the product to coexist on the same machine.

However, if you are reinstalling version 4.2 of the Java Dynamic Management Kit, you should first remove your previous installation. This procedure is described in "Removing or Reinstalling the Product Software" on page 16. Then you may proceed to reinstall the current version.

Software Packages

The Java Dynamic Management Kit 4.2 is composed of the Solaris software packages listed in Table 1–1. It is possible to install only certain packages on a machine, according to your requirements. There is a version of each package for both the Java 2 and the JDK™ 1.1 environments.

TABLE 1–1 Software and Documentation Packages

Package Name	Package Description
`SUNW1jdrt` `SUNW2jdrt`	Contains Java Dynamic Management run-time classes. Install this package on all machines where you intend to develop or run Java Dynamic Management agents and manager applications.
`SUNW1jdtk` `SUNW2jdtk`	Contains the Java Dynamic Management Kit tools (`proxygen` and `mibgen`). Install this package on all machines where you intend to develop Java Dynamic Management agents and manager applications.
`SUNW1jdex` `SUNW2jdex`	Contains the Java source code files of the supplied examples. Installing this package is optional.
`SUNW1jddc` `SUNW2jddc`	Contains all documentation for the Java Dynamic Management Kit in HTML format. Installing this package is optional.

Installation Requirements

Hardware Platforms

The Java Dynamic Management Kit runs on the Solaris Operating Environment, SPARC™ Platform Edition, and requires a minimum of 64 Mbytes of RAM.

Software

The Java Dynamic Management Kit requires the following software:

- Solaris 8 Operating Environment.
- Java 2 SDK (JDK 1.2.2) or the JDK 1.1.8 (Solaris Reference Implementations).

Disk Space

The disk space required by the software and documentation packages is as follows:

Package	Description	Disk Space for JDK 1.1 (Mbytes)	Disk Space for Java 2 (Mbytes)
`SUNW`*x*`jdrt`	Runtime classes	1.0	0.8
`SUNW`*x*`jdtk`	Toolkit classes	1.2	1.1
`SUNW`*x*`jdex`	Examples	1.3	1.3
`SUNW`*x*`jddc`	Documentation	10.5	9.5
	Total:	14.0	12.7

The installation software will compute the size of the packages you wish to install and warn you if your disk is full.

By default, the packages are installed under the `/opt` directory. The packages can be relocated if you choose the `Custom` installation. However, if you want to change the *root path* of the installation database, you must use the `pkgadd -R` *root_path* command. For information about customizing your root path, please refer to the `pkgadd(1M)` man page.

Installing the Software on a Solaris Platform

These instructions assume that you are already familiar with installing software products on machines running a Solaris operating environment, and that you are installing from a local CD-ROM drive.

The installation process uses the Solaris Web Start Wizards™ software to simplify and speed up the installation of packages on a Solaris platform. The wizard is a Java class that runs in the Java Runtime Environment, versions 1.1.8 to 1.2.2. Make sure that the `java` command is in your path before running the installer script.

You must have super-user privileges in order to install the Java Dynamic Management Kit on a Solaris platform.

Installing the Product Software

1. **Insert the product CD-ROM into the CD-ROM drive.**

   If the Volume Manager (`vold`) is running on your machine (the default), then the CD-ROM is mounted automatically under `/cdrom/jdmk_4_2` when it is loaded in the drive.

   If the Volume Manager (`vold`) is not running on your machine, create a directory called `/cdrom/jdmk_4_2` and mount the CD-ROM manually (you must be root to do this):

```
# mkdir -p /cdrom/jdmk_4_2
# mount -rF hsfs CD-RomDeviceName /cdrom/jdmk_4_2
```

   CD-RomDeviceName is the directory which represents the CD-ROM drive on your machine, for example, `/dev/dsk/c0t0d0s0`.

2. **There is a separate installer script for each version of the Java Development Kit. Type one of the following commands according to the environment in which you will use the Java Dynamic Management Kit:**

```
JDK 1.1 $ /cdrom/jdmk_4_2/solaris/11/install.sh
```

```
Java 2 $ /cdrom/jdmk_4_2/solaris/12/install.sh
```

   You may perform both installations if you wish, they are placed in distinct directories.

3. **Type the root password for your machine when prompted to do so by the installer:**

```
Solaris Web Start installer
To use this installer you will need to be the system’s root user.

Please enter this system’s root user password
Password:
```

   This will launch the interactive installer window from which you will complete the installation. If you encounter problems while using the `installer` scripts or running the installation wizard, please see “Alternate Installation Procedure” on page 15 for a workaround.

4. **Follow the instructions of the installation wizard to lead you through the installation.**

   If you choose the `Typical` installation, all of the product packages will be installed in the default location: `/opt`.

   If you choose the `Custom` installation, you can select the packages you want to install and specify a different location. Click the “Browse” button and then choose the path of the new installation directory.

   In every case, the installation wizard will automatically create an uninstallation wizard that can remove the exact configuration that was installed. See “Removing or Reinstalling the Product Software” on page 16 for details.

5. **Once the installation is finished, you may want to view or copy the documents in the** `/cdrom/jdmk_4_2/docs` **directory. These printable documents are not included in the installed packages for space considerations. See “Printable Documents” on page 32 for a description of the Java Dynamic Management Kit documentation set.**

6. **When you are finished, eject the CD-ROM.**

   If Volume Manager (`vold`) is running on your machine, the CD-ROM is unmounted automatically when it is ejected:

   ```
   $ eject cdrom
   ```

   If Volume Manager (vold) is not running on your machine, unmount the CD-ROM manually before you eject it (you must have super-user privileges to run the `umount` command):

```
# umount /cdrom/jdmk_4_2
# eject cdrom
# rmdir /cdrom/jdmk_4_2
```

Alternate Installation Procedure

If you cannot launch the installation wizard, you can simply install the software with the Solaris packages in the same directory. Follow this procedure once the CD-ROM is mounted. You must also have super-user privileges to run the `pkgadd` command.

▼ Installing with pkgadd

1. **As root, type the following command to start** pkgadd**:**

```
# /usr/sbin/pkgadd -d /cdrom/jdmk_4_2/solaris/JDKversion
```

2. **Type the numbers corresponding to the packages you want to install.**

   To install the full Java Dynamic Management Kit, including the documentation, type `all` to choose all the available packages. The installation program may prompt you with questions about package dependencies and running installation scripts. Type **y** to answer yes to all questions and complete the installation normally.

3. **When the installation is finished, type** **q** **to stop** pkgadd**.**

   You can then view the documentation or eject the CD as described in Step 5 on page 15.

Removing or Reinstalling the Product Software

You should remove the existing software packages if you are reinstalling version 4.2 of the Java Dynamic Management Kit. If you installed the packages with the installation wizard, there is a Java class file which you can launch to lead you through the uninstallation procedure.

The uninstallation wizard can only remove packages added during the previous installation. If packages remain after running the uninstallation wizard, follow the procedure in “Uninstalling with `pkgrm`” on page 17.

Note - You must have super-user privileges to remove Solaris packages, so you must be root when launching the uninstallation wizard.

▼ Running the Uninstallation Wizard

1. **Go to the location of the installed software and launch the uninstallation wizard class:**

```
# cd installDir/SUNWjdmk/jdmk4.2/JDKversion
# java uninstall_jdmk42_solaris_JDKversion
```

You may need to add the current directory (`.`) to your classpath in order to run the wizard class.

2. **Choose the type of uninstallation you wish to perform and follow the instructions.**

   A `Full` uninstall will remove all of the Java Dynamic Management Kit packages that were installed.

   A `Partial` uninstall will let you choose the packages you wish to remove. You may launch this wizard again to remove the other packages at a later date.

3. **Click the “Details” button on the last screen of the wizard to make sure the uninstallation is complete. When you are done, exit the wizard.**

   Upon exiting, the working directory will have been deleted by the uninstallation process. You will have to change to an absolute directory name. For example:

   ```
   # cd installDir
   ```

If you installed the product packages directly with the `pkgadd` command, you will have to remove them with the `pkgrm` command.

Uninstalling with `pkgrm`

1. **To find out which Java Dynamic Management Kit packages are installed on your machine, type the following command, where *x* is the major Java version number, either `1` or `2`:**

```
$ /bin/pkginfo | grep xjd
application SUNWxjddc      Java DMK online documentation
application SUNWxjdex      Java DMK examples
application SUNWxjdrt      Java DMK runtime
```

(continued)

(Continuation)

```

application SUNWxjdtk      Java DMK tools
```

2. **You must have super-user privileges to remove software packages. Call the** `pkgrm` **command with the name of the installed packages to remove them. For example, if you have a full installation of the Java Dynamic Management Kit, enter the following command, where *x* is either** `1` **or** `2`**:**

```
# /usr/sbin/pkgrm SUNWxjdtk SUNWxjdrt SUNWxjdex SUNWxjddc
```

If you are reinstalling the Java Dynamic Management Kit software, you can now follow the steps in “Installing the Product Software” on page 14.

Setting Environment Variables in a Solaris Environment

`PATH`, `CLASSPATH` and `LD_LIBRARY_PATH`

The `CLASSPATH` environment variable tells the Java virtual machine where to find the class libraries, such as the `jdmktk.jar` file.

The `PATH` environment variable specifies the location of executable files, for example, the `proxygen` tool and the `mibgen` compiler.

If needed, the `LD_LIBRARY_PATH` environment variable specifies the location of native libraries, for example, the `libstat.so` kernel statistics library.

The tools provided in the Java Dynamic Management Kit and the programs that you compile and run rely on the values in these variables. Modify these environment variables according to your Java version, as shown in Table 1–2.

TABLE 1–2 Environment Variables for the Solaris Operating Environment

Environment Variable		Add
`CLASSPATH`		*installDir*`/SUNWjdmk/jdmk4.2/1.1/lib/jdmkrt.jar:` *installDir*`/SUNWjdmk/jdmk4.2/1.1/lib/jdmktk.jar:` *installDir*`/SUNWjdmk/jdmk4.2/1.1/lib/collections.jar`
	or	*installDir*`/SUNWjdmk/jdmk4.2/1.2/lib/jdmkrt.jar:` *installDir*`/SUNWjdmk/jdmk4.2/1.2/lib/jdmktk.jar`
`PATH`		*installDir*`/SUNWjdmk/jdmk4.2/1.1/bin`
	or	*installDir*`/SUNWjdmk/jdmk4.2/1.2/bin`
`LD_LIBRARY_PATH`		*NativeLibraryPath*

CHAPTER **2**

Installation on a Windows NT Platform

The steps for installing the Java Dynamic Management Kit in a Windows NT environment are described in these sections:

- "Planning Your Installation" on page 21
- "Installing the Software on a Windows NT Platform" on page 23
- "Setting Environment Variables in a Windows NT Environment" on page 26

Planning Your Installation

Decide where to install the packages supplied with the Java Dynamic Management Kit, and ensure your machines meet the installation requirements.

Upgrading from Previous Versions

If you are upgrading your Java Dynamic Management Kit from versions 3.0, 3.2, 4.0, or 4.1, you are not required to uninstall the earlier version. Version 4.2 has a different installation structure that allows both old and new versions of the product to coexist on the same machine.

However, if you are reinstalling version 4.2 of the Java Dynamic Management Kit, you should first remove your previous installation. This procedure is described in "Removing or Reinstalling the Product Software" on page 25. Then you may proceed to reinstall the current version.

Software Packages

The Java Dynamic Management Kit 4.2 is composed of four software packages that can be installed individually. They are described in Table 2–1. It is possible to install only certain packages on a machine, according to your requirements. The software is packaged identically for the JDK 1.1 and Java 2 environments, only the contents of each package differ.

TABLE 2–1 Software and Documentation Packages

Package Name	Package Description
Runtime	Contains Java Dynamic Management run-time classes. Install this package on all machines where you intend to develop or run Java Dynamic Management agents and manager applications.
Tools	Contains the Java Dynamic Management Kit tools (`proxygen` and `mibgen`). Install this package on all machines where you intend to develop Java Dynamic Management agents and manager applications.
Examples	Contains the Java source code files of the supplied examples. Installing this package is optional.
Documentation	Contains all documentation for the Java Dynamic Management Kit in HTML format. Installing this package is optional.

Installation Requirements

Hardware

Java Dynamic Management Kit 4.2 runs on Intel platforms and requires a minimum of 64 Mbytes of RAM.

Software

The Java Dynamic Management Kit 4.2 requires the following software:

- Windows NT 4.0 operating system with Service Pack 6.
- Java 2 SDK (JDK 1.2.2) or the JDK 1.1.8 (Windows NT production releases) .

Disk Space

The following table shows the minimum disk space required for each installation.

Description	Disk Space for JDK 1.1 (Mbytes)	Disk Space for Java 2 (Mbytes)
Runtime classes	1.0	0.8
Toolkit classes	1.2	1.1
Examples	1.2	1.2
Documentation	10.1	9.3
Total:	13.5	12.4

The installation software will compute the disk space needed for the packages you choose and warn you if the installation cannot be completed.

By default, the files will be installed under the `C:\Program Files` directory, but you can specify a different path during the `Custom` installation.

Installing the Software on a Windows NT Platform

These instructions assume that you are already familiar with installing software products on machines running a Windows NT environment, and that you are installing from a local CD-ROM drive.

The installation process uses the Solaris Web Start Wizards™ software to simplify and speed up the installation of packages on a Java technology-enabled platform. It is a Java class that runs in the Java Runtime Environment, versions 1.1.8 to 1.2.2. Make sure that the `java` command is in your path before running the installer script.

▼ Installing the Product Software

1. **Insert the product CD-ROM into the CD-ROM drive.**

2. **Click the Start button on the taskbar, then choose Programs–Windows NT Explorer.**

3. **Double-click the CD-ROM icon to see the contents of the CD-ROM and open the folder named `nt`. Then choose either the `11` folder or the `12` folder according to the Java environment in which you will use the Java Dynamic Management Kit.**

   You may perform both installations if you wish, they are placed in distinct directories.

4. **Open the chosen folder and double-click the `setup.bat` filename or icon.**

   This will launch the interactive installer window from which you will complete the installation. If you encounter problems while using the `setup.bat` scripts or running the installation wizard, please see "Alternate Installation Procedure" on page 24 for a workaround.

5. **Follow the instructions of the installation wizard to lead you through the installation.**

   If you choose the `Typical` installation, all packages will be installed in the default location: `C:\Program Files`.

   If you choose the `Custom` installation, you can select the packages you want to install and specify a different location. Click the "Browse" button and then choose the path of the new installation directory.

   In every case, the installation wizard will automatically create an uninstallation wizard that can remove the exact configuration that was installed. See "Removing or Reinstalling the Product Software" on page 25 for details.

Note - The installation wizard doesn't update the Windows NT registry with information about the installed packages, it only unpacks and moves files to your file system.

---

6. **Once the installation is finished, you may want to view or copy the documents in the `docs` folder at the root of the CD-ROM. These printable documents are not included in the installed packages for space considerations. See "Printable Documents" on page 32 for a description of the Java Dynamic Management Kit documentation set.**

7. **When you are finished, eject the CD-ROM.**

Alternate Installation Procedure

If you cannot launch the installation wizard, you will have to manually copy the product files from the CD-ROM to your disk. Follow this procedure to find the files and install them properly:

Installing from the ZIP file

1. **Open the `other` folder at the root of the CD-ROM**

2. **Extract the contents of the `jdmk42.zip` file to the *installDir* folder of your choice.**

   This file contains the whole file hierarchy of a full installation for both the JDK 1.1.*x* and Java 2 environments. Either delete the branches of this hierarchy that you don’t need, or use an archive utility to extract only the files you need.

   Note - This archive is not platform specific and therefore doesn’t contain the scripts for launching `proxygen` and `mibgen` in its `bin` directories. Instead, you have to launch these tools from their corresponding Java class: `com.sun.jdmk.tools.ProxyGen` and `com.sun.jdmk.tools.MibGen`.

Removing or Reinstalling the Product Software

You should remove the existing software packages if you are reinstalling version 4.2 of the Java Dynamic Management Kit. If you installed the packages with the installation wizard, there is a Java class file which you can launch to lead you through the uninstallation procedure. Make sure that your classpath is set up properly for your version of the `java` command.

Note - Because the packages of the Java Dynamic Management Kit are not listed in the Windows NT registry, you must go through the uninstaller to remove the software. If you installed the software from the ZIP file, you will need to delete the files of the installation by hand.

Running the Uninstallation Wizard

1. **Open a command prompt window and go to the location of the installed software and launch the uninstallation wizard class:**

```
C:\>cd "installDir\SUNWjdmk\jdmk4.2\JDKversion"
C:\>java uninstall_jdmk42_nt_JDKversion
```

You may need to add the current directory (.) to your classpath to run the wizard class.

2. **Choose the type of uninstallation you wish to perform and follow the instructions.**

   A `Full` uninstall will remove all of the Java Dynamic Management Kit packages that were installed.

   A `Partial` uninstall will let you choose the packages you wish to remove. You may launch this wizard again to remove the other packages at a later date.

3. **Click the "Details" button on the last screen of the wizard to make sure the uninstallation is complete. When you are done, exit the wizard.**

Note - The uninstallation wizard can only remove packages added during the previous installation. If packages remain after running the uninstallation wizard, you will need to delete their files by hand.

Setting Environment Variables in a Windows NT Environment

PATH and CLASSPATH

The CLASSPATH environment variable tells the Java virtual machine where to find the class libraries, such as the `jdmkrt.jar` file.

The PATH environment variable specifies the location of executable files, for example, the `proxygen` tool and the `mibgen` compiler.

The tools provided in the Java Dynamic Management Kit and the programs that you compile and run rely on the values in these variables. Modify these environment variables according to your Java version, as shown in Table 2–2.

TABLE 2–2 Environment Variables for the Windows NT Environment

Environment Variable	Add
`CLASSPATH`	*installDir*`\SUNWjdmk\jdmk4.2\1.1\lib\jdmkrt.jar;` *installDir*`\SUNWjdmk\jdmk4.2\1.1\lib\jdmktk.jar;` *installDir*`\SUNWjdmk\jdmk4.2\1.1\lib\collections.jar`
or	*installDir*`\SUNWjdmk\jdmk4.2\1.2\lib\jdmkrt.jar;` *installDir*`\SUNWjdmk\jdmk4.2\1.2\lib\jdmktk.jar`
`PATH`	*installDir*`\SUNWjdmk\jdmk4.2\1.1\bin`
or	*installDir*`\SUNWjdmk\jdmk4.2\1.2\bin`

CHAPTER **3**

Release Notes

These release notes contain:

- "Release and Version Issues" on page 29
- "Overview of the Product Documentation" on page 31
- "Working Directories" on page 33
- "Known Limitations" on page 33

Release and Version Issues

JMX Specification Compatibility

The Java Dynamic Management Kit version 4.2 is compatible with the JMX Instrumentation and Agent Specification, v1.0 (Final Release, July 2000). This is the current release of the JMX specification at the time the Java Dynamic Management Kit is shipping (December 2000).

The corresponding document can be viewed online once the documentation package has been installed (see "Overview of the Product Documentation" on page 31).

Java Version Compatibility

The Java Dynamic Management Kit version 4.2 is officially supported for development platforms under the following configurations:

- Solaris 8 Operating Environment (SPARC Platform Edition), with Java 2 SDK, v1.2.2 Reference Implementation

- Windows NT 4.0 service pack 6 on Intel platform with Java 2 SDK, v1.2.2 Production Release for Windows NT

Deployment platform configurations are unsupported. However, the Java Dynamic Management Kit components are 100% Java code and should run with minimal modifications on environments compliant with Java Runtime Environment for Java 2, v1.2.2 or JDK 1.1.8. Many other configurations have been tested and are known to work reliably.

The Java Dynamic Management Kit 4.2 has been tested with Java 2 SDK (JDK 1.3.0) and no problems were encountered. JDK 1.1.x versions are provided *as is* but are not officially supported.

Backwards Compatibility

Versions 4.2 and 4.1 of the Java Dynamic Management Kit are fully communications-compatible: applications developed and running with version 4.1 can communicate with those developed and running 4.2. There is complete compatibility between agents using 4.1 and managers using 4.2. In the opposite case of an agent using 4.2 and a manager using 4.1, there is only a minor limitation with the connector heartbeat feature (see "Heartbeat Cross-Compatibility" on page 34).

Versions 4.2, 4.1, and 4.0 of the Java Dynamic Management Kit are fully code-compatible: applications developed with version 4.1 and 4.0 will compile without modification when using version 4.2 of the packages. Some classes may cause compiler warnings about deprecation, but functionality is unchanged.

Due to major architectural changes, including the JMX compatibility, there is no communications-compatibility between version 4.2 of the Java Dynamic Management Kit and versions 4.0, 3.2, 3.0, and 2.0. Code developed for versions 3.2, 3.0, and 2.0 will most likely need to be ported in order to take advantage of the new architecture. Agents and managers developed with different versions of the product are definitely incompatible at the communications level, except for SNMP.

Due to its relatively stable protocol definition, SNMP can be used to connect applications developed with different versions of the Java Dynamic Management Kit, to the extent that the protocol was implemented by those versions. Agents or managers must continue to run their original version of the product, but they may communicate through SNMP with entities developed and running with different versions.

Files generated by the mibgen tool of the 4.2 release are compatible with those generated by the 4.1 version with one exception. The metadata classes generated for a MIB and used by the SNMP adaptor need to be regenerated. The other classes generated to represent the whole MIB and a skeleton implementation of each MIB group are unchanged, and they can be safely replaced by your previously customized implementations.

Cross-Java Compatibility

You must run all communicating agents and managers with the same major version of the Java Runtime Environment, either all using JDK 1.1 or all using Java 2. The behavior of a manager running Java 2 and communicating with an agent running the JDK 1.1, or vice-versa, is undefined. Data integrity and behavioral integrity are only possible when both agent and manager run the same version of the Java environment.

Again, SNMP interoperability has the advantage over other protocols of being independent of the Java environment version. The SNMP protocol adaptor can, regardless of its Java version, receive and decode requests from any SNMP manager. And a manager implemented with the SNMP manager API can send requests to any agent, without needing to know about its version compatibility.

Cross-Platform Compatibility

When product version compatibility and Java version compatibility are respected, the Java Dynamic Management Kit offers full management compatibility across heterogeneous platforms. For example, a manager running on a SPARC server could manage a whole deployment of Windows NT PCs, controlling their availability and upgrading their Java-technology based software when needed.

Overview of the Product Documentation

The Java Dynamic Management Kit product includes both printable and online documentation, as well as an exhaustive set of programming examples.

Online HTML Files

The HTML documentation is available after an installation of the product that includes the documentation package. On the machine where you installed the product, open the following URL in any browser:

In a Solaris environment: `file:/opt/SUNWjdmk/jdmk4.2/`*JDKversion*`/index.html`

On Window NT: `file:/Program Files/SUNWjdmk/jdmk4.2/`*JDKversion*`/index.html`

This page contains links to all the product documentation supplied online with the Java Dynamic Management Kit, including:

- *Getting Started with the Java Dynamic Management Kit 4.2*

- *Java Dynamic Management Kit 4.2 Tools Reference*
- *Java Dynamic Management Kit 4.2 Tutorial*
- *Java Dynamic Management Kit 4.2 Installation Guide and Release Notes* (this document).
- The Java Dynamic Management Kit API specification generated by Javadoc™; the separate Collections API is also included in the JDK 1.1 version of the product.
- The JMX specification document.

Printable Documents

Complete PostScript™ and PDF versions of the above books are supplied on the CD-ROM of the product. These files are located in the `docs` directory at the common root of the CD-ROM.

The documents are formatted for US Letter paper size (8.5 × 11 inches), but the they can be loaded by any appropriate document viewer or printed directly to any printer, regardless of the default paper size. The text area on each page will fit on all standard paper sizes.

Programming Examples

Sample applications which demonstrate most of the functionality of the Java Dynamic Management Kit are provided in the examples package of the product. If you installed this package, the Java source files and `README` text files for these applications are located in subdirectories under:

installDir`/SUNWjdmk/jdmk4.2/`*JDKversion*`/examples`

Note - On the Solaris platform, writing to this directory requires super-user privileges. In order to compile the example programs, users should copy the examples hierarchy to a more accessible location.

The `README` file for each example gives a brief explanation of the source files and the instructions for running its application. Further explanation for most examples is given in the Java Dynamic Management Kit 4.2 Tutorial.

The `examples` directory also contains the `JdmkProxyMBeans` subdirectory which provides proxy MBeans for all of the Java Dynamic Management Kit components that support them. These are generated by the `mibgen` tool and will need to be compiled in the normal way before use. They can then be used to provide proxy objects in your manager applications for the agent-side service MBeans.

Working Directories

The installation of the Java Dynamic Management Kit creates directories that certain component services use by default. These directories are located under the Java version-specific branch of the installed directories, meaning that both Java versions of the product can be installed and run simultaneously. The following table shows the name and usage of these working directories (we show only Solaris path names, Windows NT folder names are identical):

TABLE 3–1 Working Directories

Directory	Description
installDir`/SUNWjdmk/jdmk4.2/`*JDKversion*`/etc/conf`	Contains the `template.acl` file and is the default location for the `jdmk.acl` file for defining access rights in SNMP agents
installDir`/SUNWjdmk/jdmk4.2/`*JDKversion*`/etc/mibgen`	Contains the `mib_core.txt` file that is always used by default when "compiling" MIBs with the `mibgen` tool
installDir`/SUNWjdmk/jdmk4.2/`*JDKversion*`/tmp`	A storage area for native libraries loaded by the m-let service

If you install the Java Dynamic Management Kit in the default directory on a Solaris platform, *installDir* will be `/opt`. In this case, you will need super-user privileges to write the `jdmk.acl` file in the `etc/conf` directory.

Known Limitations

The following issues are known limitations of the Java Dynamic Management Kit 4.2 product at the time this document is being written (October 2000). Here, we explain the observed inconsistencies and suggest a workaround:

Heartbeat Cross-Compatibility

When a manager application running with version 4.1 of the product is communicating with agents running version 4.2, the heartbeat mechanism will cause an exception the second time the `connect` method is called. This behavior occurs in all connectors: RMI, HTTP, and HTTPS.

If you do not need the heartbeat functionality, you may avoid this limtation by disabling the heartbeat. If you do so, you must disable the heartbeat *before the first connection*, by setting the heartbeat period to 0 (zero). Once you have disabled the heartbeat, your connector client may connect and disconnect as many times as you wish.

If you would like to use the heartbeat functionality in this configuration, you should not establish more than one connection per connector client. The heartbeat mechanism is fully functional during the first connection. If you need establish a second connection after the first, you should disconnect and instantiate a new connector client for the second connection.

SNMPv2 Protocol

`Report` PDUs are not supported in the Java Dynamic Management Kit 4.2 because they are not presently defined by the standards (RFC 1905). `Trap` PDUs (both v1 and v2) and `InformRequests` are fully supported and should be used to send management information between agents and managers or between managers, respectively.

The HTML Adaptor

Due to the limitations of the HTML protocol, the HTML adaptor does not provide the same management view as a connector. For example, you cannot set attributes individually in the MBean view: all setters are called every time the “Apply” button is selected, even if the content hasn’t changed.

Also, the HTML adaptor can only represent a limited set of types through an HTML interface. The values of attribute or operation results which rely on an unsupported type cannot be displayed or entered by the user. This means that some component functionality is inaccessible through the HTML adaptor.

HTTPS Heartbeat on Windows NT

The HTTPS connector client on the Windows NT platform may signal a false `CONNECTION_REESTABLISHED` notification, when its connector server is stopped.

Thereafter, the correct `CONNECTION_LOST` notification is received. As this is due to timing issues, it is best to use a heartbeat period of five seconds or longer.

RMI Registry

Before starting a second RMI adaptor server in the same agent, you must launch an RMI registry for its intended port. Use the `rmiregistry` command line tool to do this, as explained in the JDK Tools online documentation (*JDKinstallDir*`/docs/tooldocs/`*operatingEnv*`/rmiregistry.html`). Failure to do so will prevent the second RMI adaptor server from starting.

Tutorial

The tutorial doesn’t cover all of the sample applications found in the `examples` directory. All of the examples include detailed `README` files and well-commented code. In addition, certain topics can be found elsewhere:

- The MBean registration process is covered in the JMX agent specification.
- The monitoring and timer services are covered in the JMX specification.
- SNMP UserData is covered in the programming examples.
- SNMP RowStatus is covered in the programming examples.

JDK 1.1.8 and Native Threads

When running the JDK 1.1.8 virtual machine with native threads and using an HTTP connector client, an unexpected `java.lang.ClassCastException` is sometimes thrown. The workaround to this problem is to deactivate the keep-alive feature of the `java.net.HttpURLConnection` class. You can do this by running your application with the following property definition on the command line:

```
$ java -Dhttp.keepAlive=false yourApplication
```

Green Threads

Using *green* threads when running with a Java virtual machine other than a production release is also a known source of errors. Avoid using this combination to minimize the risk of unexplained failures. Please refer to the JDK documentation for more information about native and green threads.

M-Let with JDK 1.3 Software

When using the m-let service with the JDK 1.3 software, in particular when calling the method getMBeansFromURL(java.lang.String url) or getMBeansFromURL(java.net.URL url), a javax.management.ServiceNotFoundException is raised. This occurs because the JDK 1.3 APIs send HTTP/1.1 client requests and some HTTP server versions will reject all client requests that are not HTTP/1.0 or HTTP/0.9. Therefore, you need to ensure that your HTTP server accepts HTTP/1.1 client requests.

SnmpVarBind Status

In prior releases of the Java Dynamic Management Kit, the SnmpVarBind *status* was handed differently, depending upon whether you were using SNMP v1 or SNMP v2. This is no longer the case with the Java Dynamic Management Kit 4.2. The SnmpVarBind *status* handling has been modified to allow the user to perform status checking in the same way for both SNMP v1 and SNMP v2, although it may impact the Java code if you are using SNMP v2.

Now, regardless of the SNMP version you use to send a request, theSnmpVarBind status, as given by the getValueStatus method, always reflects the way that the SNMP request was sent (i.e. whether you used SNMP v1 or SNMP v2). Therefore, you may compare it with the constants stValueOk, stValueEndOfMibView, stValueNoSuchInstance, stValueNoSuchObject, or stValueUnspecified, which are defined in the SnmpVarBind class.