# *NEXUS*

### *Version 1.2*

### *ADMINISTRATOR DOCUMENTATION*

***Christophe Cartwright***

***IT DEVELOPER***

**TABLE OF CONTENTS**

[Documentation Overview 2](#_dfqchmm3lugq)

[Java Virtual Machine (JVM) Installation 3](#_ezjbjf23ekvp)

[Installation Instructions 3](#_jw0pnlmwtl7n)

[Installed Directory Tree 4](#_yq9e9yg7kl7u)

[Uninstalling the Java 2 SDK 6](#_8xxfr3v5rhd8)

[Installation of JDBC Drivers 7](#_if46uvr6z0ta)

[SQL Server 2000 Driver for JDBC (Version 2.2.0029) 7](#_vbndfipbswm6)

[Setting the ClassPath 7](#_xzciwm9pk3je)

[Installation on Windows 8](#_azfsw18679kk)

[Uninstalling 9](#_qeuxjfkjgvi)

[Using the MSSQL Server 7.0 Driver for JDBC (JTDS JDBC Driver) 9](#_amg92xhjnlc2)

[Setting the Classpath 9](#_vzpkg6ak1j99)

[Using the Oracle Driver for JDBC (Driver Version JDBC Thin Driver 9.0.1) 10](#_lphvgeiyc2c1)

[The Driver for Oracle9i JDBC (Version 2.2.0029) can be found at Oracle’s website: 10](#_i6vw8txkd3hf)

[Installation 10](#_m8a1yjyatuee)

[Setting the Classpath 10](#_ivu8cjgoi79e)

[NEXUS Module as a Service 11](#_4ekh3671pzb)

[Creating the NEXUS Jar files: 11](#_r7yig3emrac5)

[NEXUS Module: 11](#_c8k9z329dl8a)

[NEXUS Jar File 11](#_7hcmem6yf7w0)

[Java Service Wrapper 13](#_74gbu3ysvq5y)

[Download and Installation 13](#_cnk1xj26hpxh)

[Application Integration with the Java Service Wrapper 13](#_2d6ne3fxlnbf)

[Installing wrapper files 15](#_x9syzwsbfycn)

[Bin Directory 15](#_6kyiu1vxef8o)

[Lib Directory 15](#_atbf4leyn9ug)

[The Conf Directory 16](#_31eopq4fnm94)

[Logs Directory 16](#_2eltjedi805)

[Application Command Line: Profile Module 16](#_rxqw846t7x1t)

[Modifying the Nexus\_Run.bat file 17](#_fxx9zi79s5g1)

[Modifying the warpper.conf files 17](#_i2q89nu8m78w)

[Java Executable 17](#_25zhf19pmupl)

[Java Arguments 18](#_jjcmr1gyvdit)

[Java Classpath 18](#_fxl3jbmj2dbj)

[Java Main Class 18](#_p5cj7h2ynoyq)

[Library Path 19](#_s0lcz3a0w20v)

[Wrapper NT/2000/XP Service Properties 19](#_9dumvpbggum2)

[Launching the Services 19](#_lk5sjvoa063i)

# 

# **Documentation Overview**

| Manual | Version | Contents |
| --- | --- | --- |
| Java Virtual Machine (JVM) | 1.1 | Deals with detail installation of the Java Virtual machine (JVM) and java database connectivity (JDBC) drivers |
| Java Database Connectivity Drivers (JDBCD) | 1.1 | Deals with detail installation of the Java database connectivity (JDBC) drivers |
| NEXUS Module as a Service | 1.1 | Details the setup procedure for establishing the NEXUS Module as an NT service. |

# **Java Virtual Machine (JVM) Installation**

The current JVM version used to run the NEXUS Module is:

**Java(TM) 2 SDK, Standard Edition Version 1.4.2\_11**

The Java 2 SDK is a development environment for building applications, applets, and components that can be deployed on the Java platform. The Java 2 SDK software includes tools useful for developing and testing programs written in the Java programming language and running on the Java platform. These tools are designed for use from the command line. Except for applet-viewer, these tools do not provide a graphical user interface.

## ***Installation Instructions***

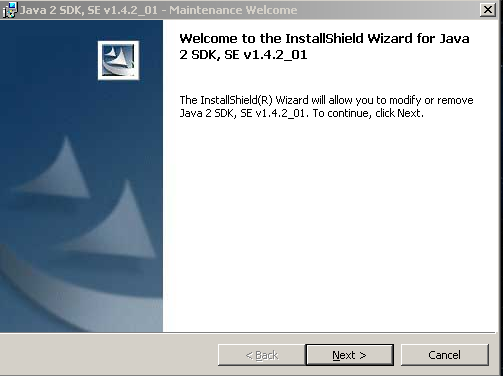
In this procedure, you will run the self-installing executable to unpack and install the Java 2 SDK software bundle. As part of the Java 2 SDK, this installation includes the Java Plug-in and Java Web Start, as well as an option to include the public Java 2 Runtime Environment. (The Java 2 SDK also contains a [private](http://java.sun.com/j2se/1.4.2/#private) J2RE for use only by its tools.)

1. Check the download file size **(Optional)** 
   * If you save the self-installing executable to disk without running it from the download page at the web site, notice that its byte size is provided on the download page. Once the download has completed, check that you have downloaded the full, uncorrupted software file.
2. If 1.4.2 Beta is installed, uninstall it.

* Use the Microsoft Windows Add/Remove Programs utility, accessible from the Control Panel (Start -> Settings -> Control Panel).

1. Run the Java 2 SDK installer

* *Note - you must have administrative permissions in order to install the Java 2 SDK on Microsoft Windows 2000 and XP.*



The file j2sdk-1\_4\_2\_<version>-windows-i586-i.exe is the Java 2 SDK installer. If you downloaded it instead of running it directly from the web site, double-click on the installer's icon. Then follow the instructions the installer provides. The installer may ask you to reboot your computer. When done with the installation, you can delete the download file to recover disk space.

### **Installed Directory Tree**

The Java 2 SDK has the directory structure shown below.

j2sdk1.4.2\_<version>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

| | | | bin lib | demo |

| | | LICENSE | | | jre

| | COPYRIGHT | \_\_|\_\_

| README.txt include | |

readme.html bin lib

4.Update the PATH variable **(Optional)**

* You can run the Java 2 SDK without setting the PATH variable, or you can optionally set it as a convenience.
* **Should I set the PATH variable?**  
  Set the PATH variable if you want to be able to conveniently run the Java 2 SDK executables (javac.exe, java.exe, javadoc.exe, etc.) from any directory without having to type the full path of the command. If you don't set the PATH variable, you need to specify the full path to the executable every time you run it, such as:

<**Drive>**:**\j2sdk1.4.2\_<version>\bin\javac MyClass.java**

* **How do I set the PATH permanently?**   
  To set the PATH permanently, add the full path of the **j2sdk1.4.2\_<version>\bin** directory to the PATH variable. Typically this full path looks something like:<**Drive>:\sdk1.4.2\_<version>\bin**.

Set the PATH as follows, according to whether you are on Microsoft Windows NT or 98/2000/ME.

**Microsoft Windows NT, 2000, and XP** - To set the PATH permanently:

Choose:

Start -> Settings -> Control Panel

Double-click System.

* + - On Microsoft Windows NT, select the Environment tab.
    - On Microsoft Windows 2000 select the Advanced tab.

Select Environment Variables.

* + - Look for "Path" in the User Variables and System Variables.
    - If you're not sure where to add the path, add it to the right end of the "Path" in the User Variables.
    - A typical value for PATH is: <**Drive>:\2sdk1.4.2\_<version>\bin**
    - Capitalization doesn't matter. Click "Set", "OK" or "Apply".

The PATH can be a series of directories separated by semi-colons (;). Microsoft Windows looks for programs in the PATH directories in order, from left to right. You should only have one **bin** directory for a Java SDK in the path at a time.

The new path takes effect in each new Command Prompt window you open after setting the PATH variable.

5. Start using the Java 2 SDK!

* Your computer system should now be ready to use the Java 2 SDK. In this step, you'll run some simple commands to make sure it is working properly.
* If you are new to developing and running programs in the Java programming language, see [The Java Tutorial](http://java.sun.com/docs/books/tutorial/) online for some guidance. Note especially the tutorial trails under the heading *Trails Covering the Basics*.
* You can also [download the Java 2 SDK documentation from the Java 2 SDK download page.](http://java.sun.com/j2se/1.4.2/index.html).

## ***Uninstalling the Java 2 SDK***

If you should ever want to uninstall the Java 2 SDK, use the "Add/Remove Programs" utility in the Microsoft Windows Control Panel. As an alternative method, if you still have the original installation program that you used to install the Java 2 SDK, you can double click on it to launch an uninstall program.

# **Installation of JDBC Drivers**

## ***SQL Server 2000 Driver for JDBC (Version 2.2.0029)***

The Type 4 SQL Server 2000 Driver for JDBC provides JDBC access through any Java-enabled applet, application, or application server.

Before installing the SQL Server 2000 Driver for JDBC, verify that SQL Server 2000 installed on the server one will connect to and that the system containing the JDBC Driver meets requirements (see below).

*Product Requirements for SQL Server 2000 Driver for JDBC*

* JDBC 1.22 API JRE 1.1.8 or higher
* JDBC 2.0 Core API JRE 1.2 or higher
* JDBC 2.0 Optional Package JRE 1.2 or higher

### **Setting the ClassPath**

The SQL Server 2000 Driver for JDBC needs to be defined in your CLASSPATH variable. If the drivers are not on your CLASSPATH, you receive the error "class not found" when trying to load the driver.

Set your system CLASSPATH to include the following entries, where *install\_dir* is the path to your SQL Server 2000 JDBC Driver installation directory.

*Ex (*install\_dir = ***<drive>:\******j2sdk1.4.2\_<version>\jre\lib\ext)***

The following are the Jar files one needs to install:

* ***install\_dir*/msbase.jar**
* ***install\_dir*/msutil.jar**
* ***install\_dir*/mssqlserver.jar**

**Windows Example**

* CLASSPATH=
  + <drive>:\Microsoft SQL Server 2000 Driver for JDBC\lib\msbase.jar;
  + <drive>:\Microsoft SQL Server 2000 Driver for JDBC\lib\msutil.jar;
  + <drive>:\Microsoft SQL Server 2000 Driver for JDBC\lib\mssqlserver.jar

**UNIX Example**

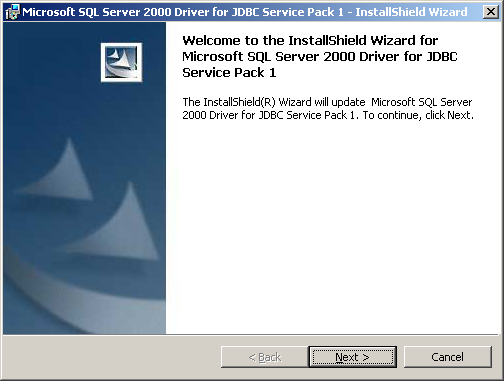
* CLASSPATH=
  + /home/user1/mssqlserver2000jdbc/lib/msbase.jar;
  + /home/user1/mssqlserver2000jdbc/lib/msutil.jar;
  + /home/user1/mssqlserver2000jdbc/lib/mssqlserver.jar

### **Installation on Windows**

The Microsoft SQL Server 2000 Driver for JDBC includes a Setup program that enables you to install the product with a graphical user interface.

Before you start installation, exit or close all applications that are running to prevent file-locking conflicts. In addition, verify that your system meets the specified requirements (see above - System Requirements).

**1.** Double-click the Setup program. The Welcome window appears.



**2.** On the Welcome window, click **Next** to continue. The License Agreement window appears. NOTE: You can exit the Setup program at any time by clicking **Cancel**, or return to the previous window by clicking **Back**.

**3.** On the License Agreement window, select the appropriate option. If you accept the agreement, click **Next**. If you do not accept the agreement, click **Cancel** and exit the installation.

**4.** On the Setup Type window, select Complete or Custom. After making a selection, click **Next**. If you choose Complete, all features are installed in the default directory (usually, <drive>:\Program Files\SQL Server 2000 Driver for JDBC). Skip to Step 6. If you choose Custom, you can elect not to install the online books, and can select an installation directory. Proceed to Step 5.

**5.** There are two options that you can change on the Custom Setup window. After making your selections, click **Next**. If you do not want to install the online books, click the Online Books icon and select the appropriate option. If you do not want to install to the default directory, click **Change** and choose another installation directory.

**6.** On the Ready to Install window, verify that the information is correct. Click **Back** to make any corrections or click **Next** to install.

**7.** When the installation has finished, the Setup Completed window appears. Click **Finish** to exit the Setup program.

### **Uninstalling**

You can uninstall the SQL Server 2000 Driver for JDBC through the Add/Remove Programs option in the Control Panel of the Windows Start menu. After opening Add/Remove Programs, select SQL Server 2000 Driver for JDBC and click **Remove**.

## ***Using the MSSQL Server 7.0 Driver for JDBC (JTDS JDBC Driver)***

### **Setting the Classpath**

The SQL Server 7.0 Driver for JDBC needs to be defined in your CLASSPATH variable. If the drivers are not on your CLASSPATH, you receive the error "class not found" when trying to load the driver.

Set your system CLASSPATH to include the following entries, where *install\_dir* is the path to your SQL Server 7.0 JDBC Driver installation directory.

*Ex (*install\_dir = ***<drive>:\******j2sdk1.4.2\_<version>\jre\lib\ext)***

The following is the Jar files one needs to install:

* ***install\_dir*/jtds-0.5.1.jar**

## ***Using the Oracle Driver for JDBC (Driver Version JDBC Thin Driver 9.2.0.8)***

### The Driver for Oracle9i JDBC (Version 9.2.0.8) can be found at Oracle’s website:

[***https://otn.oracle.com/software/tech/java/sqlj\_jdbc/index.html***](http://otn.oracle.com/software/tech/java/sqlj_jdbc/index.html)

Oracle's JDBC Thin driver is a Type 4 driver that uses Java sockets to connect directly to Oracle. It provides its own implementation of a TCP/IP version of Oracle's Net. Because it is written entirely in Java, this driver is platform-independent.

### **Installation**

* *After downloading the driver you should have a jdbc\_thin\_9\_0\_1 winzip file.*
* *Extracting the file will produce 20 files. It is the*
  + **ocrs12.jar**
  + **nls\_charset12.jar**
  + **classes12.jar**

which are needed to make a JDBC connection to Billing.

### **Setting the Classpath**

The Oracle9i Driver for JDBC needs to be defined in your CLASSPATH variable. If the drivers are not on your CLASSPATH, you receive the error "class not found" when trying to load the driver.

Set your system CLASSPATH to include the following entries, where *install\_dir* is the path to Oracle9i for JDBC installation directory.

Ex (*install\_dir =*  <drive>:\ j2sdk1.4.2\_<version>\jre\lib\ext)

The following is the Jar files one needs to install:

* ***install\_dir*/ ocrs12.jar**
* ***install\_dir*/ nls\_charset12.jar**
* ***install\_dir***/ **classes12.jar**

These files contain the Oracle9i Driver to connect to the Billing Database. Please do not try to put multiple versions of the Oracle JDBC drivers In the CLASSPATH.

# **NEXUS Module as a Service**

## ***Creating the NEXUS Jar file:***

The NEXUS Java archive (Jar) file is the java executable file containing the java class files and packages that compose the NEXUS Module. Below is the list of the Java class files and the package:

### **NEXUS Module:**

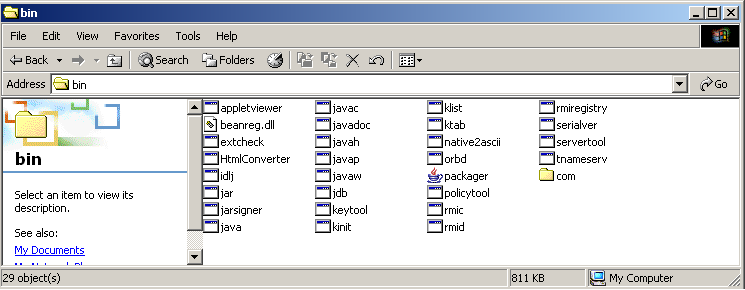
* **NEXUS Package:** 
  + com.<relevant\_path>.nexus

Within the NEXUS package are the class files for the NEXUS module.

* **NEXUS Classes:**
  + \_NEXUS\_JDBC\_Connection.Class
  + \_NEXUS\_Module.Class
  + \_NEXUS\_Run.Class
  + \_NEXUS\_Thread.Class

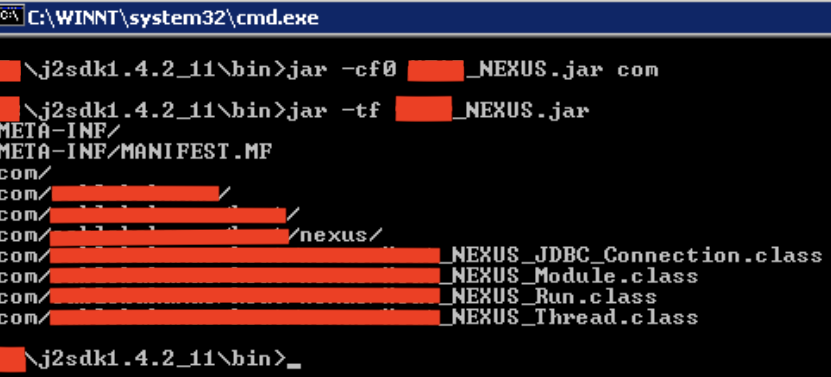
### **\_NEXUS Jar File**

* Copy the com directory to the \bin directory of your JVM, ex: <drive>:\j2sdk1.4.2\_11\bin

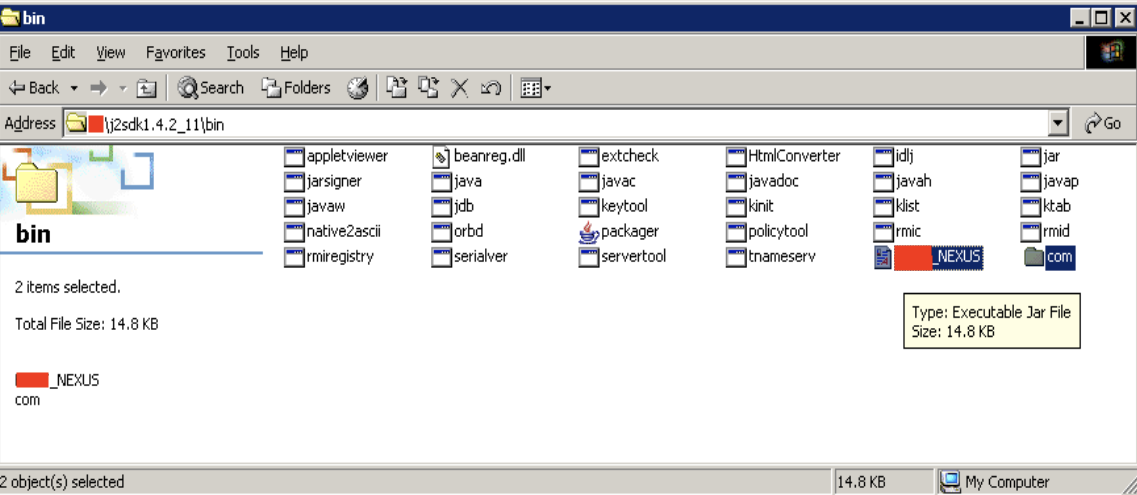


* Run the MS-Dos console and go to the \bin directory of your JVM and type the following: **jar -cf0 \_NEXUS.jar com**
* This will create the \_NEXUS.jar file.
  + The –cf is the create file command.
  + The 0 specifies that the files in the jar file will remain uncompressed (for faster execution).
  + The<file\_name>.jar is the name of the JAR file name.
  + The com indicates the package which contains all the class files to run the
* To confirm that the correct class files and package were bundled to create the correct jar file type the command: **jar -tf \_NEXUS.jar**

After this command, you should get the following at the MS-Dos console:



* Within the JVM \bin directory you should now have a Jar file named **\_NEXUS.jar**:



## ***Java Service Wrapper***

The Java Service Wrapper is an application that has evolved out of a desire to solve a number of problems common to many Java applications. The Wrapper makes it possible to install a Java Application as a Windows NT Service.

The scripts provided with the Wrapper also make it very easy to install those same Java Applications as daemon processes on UNIX systems. The Wrapper correctly handles user log outs under Windows, service dependencies, and the ability to run services that interact with the desktop.

For additional information, please visit:

[https://wrapper.tanukisoftware.org/doc/english/index.html](http://wrapper.tanukisoftware.org/doc/english/index.html)

### **Download and Installation**

* Download the wrapper from: [https://wrapper.tanukisoftware.org/doc/english/index.html](http://wrapper.tanukisoftware.org/doc/english/index.html)
* Extract the winzip file. This will give the directory “wrapper\_win32\_3.0.4”

### **Application Integration with the Java Service Wrapper**

* Here the WrapperSimpleApp helper class is used to launch the application. This is the simplest way to integrate with the Wrapper, and where possible, it is highly recommended.
* There are some things to be aware of when using this method. When the Wrapper shuts down the JVM, there is no direct call to an application requesting that it shutdown cleanly. Rather, the Wrapper will exit the JVM by calling System.exit() from within the JVM.
* If the application has registered its own shutdown hook, it will be invoked, giving the application a chance to shutdown cleanly.
* If on the other hand, a shutdown hook is not registered, then the application will suddenly exit.
* Both cases, with and without a shutdown hook, provide the exact same behavior as if the application was running without the Wrapper and a user pressed CTRL-C in the console.
* When integrating with the WrapperSimpleApp helper class, the WrapperSimpleApp class replaces an application's main class.
* This gives the WrapperSimpleApp class a chance to immediately initialize the WrapperManager and register the JVM with the Wrapper. The WrapperSimpleApp class then manages all interaction with the Wrapper as well as the life-cycle of an application.
* When the Wrapper sends a start message to the JVM via the WrapperManager, the main method of the application's actual main class is called.
* The WrapperSimpleApp helper class is told how to launch the application by passing the application's main class name, followed by any additional application parameters to the main method of the WrapperSimpleApp.

### **Installing wrapper files**

* Locate the directory that currently has the java source and class files associated with the Nexus Module.
* Create a nexus directory in the above directory. Within this directory create bin, conf, lib and log directories.

### **Bin Directory**

* Copy the following files into the {nexus}\bin directory:
  + {WRAPPER\_HOME}\bin\Wrapper.exe
  + {WRAPPER\_HOME}\src\bin\App.bat.in
  + {WRAPPER\_HOME}\src\bin\InstallApp-NT.bat.in
  + {WRAPPER\_HOME}\src\bin\UninstallApp-NT.bat.in
* Rename the three batch files in each directory (as shown below). Be sure to remove the .in extensions so that the files all end in .bat. Depending on how your file explorer is configured, you may not be able to see the extensions.
  + {nexus}\bin\\_NEXUS\_App.bat
  + {nexus}\bin\Install\_NEXUS\_App-NT.bat
  + {nexus}\bin\Uninstall\_NEXUS\_App-NT.bat
* The Wrapper.exe file is the actual Wrapper executable.
* The batch files are used to run the \_NEXUS\_Module in a console, and to install and remove them as an NT Services. These scripts should not require any modification.
* They assume that the wrapper.conf file will be located within a conf directory one level up, ../conf/wrapper.conf.
* If you wish to locate this file someplace else, then the batch files will require that small modification.

### **Lib Directory**

* Copy the following two files into the {nexus}\lib directory:
  + {WRAPPER\_HOME}\src\lib\wrapper.dll
  + {WRAPPER\_HOME}\src\lib\wrapper.jar
* The Wrapper.DLL file is a native library required by the portion of the Wrapper that runs within the JVM. The wrapper.jar file contains all of the Wrapper classes.

### **The Conf Directory**

The Wrapper requires a configuration file. The standard location for this file is in a conf directory in the application's home directory.

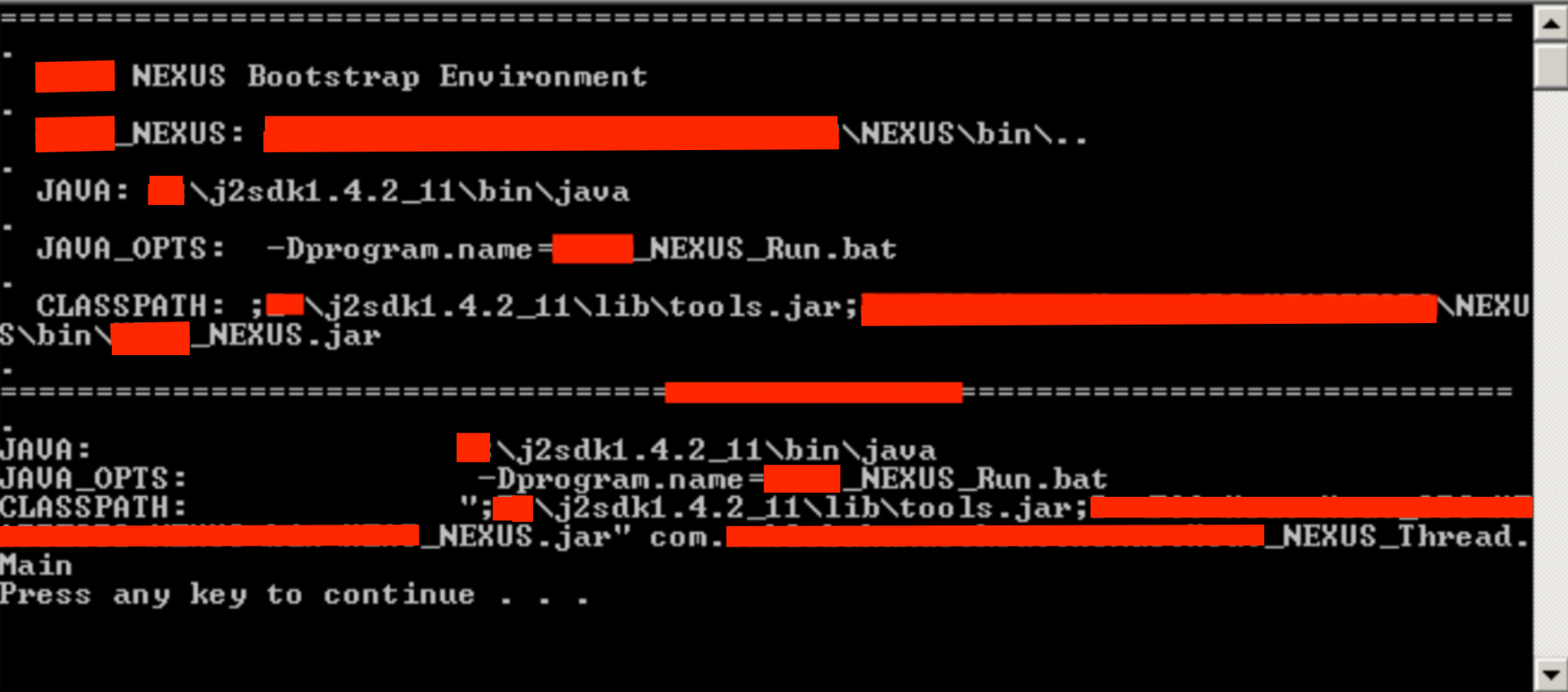
* Copy the template wrapper.conf file from: {WRAPPER\_HOME}\conf\wrapper.conf.ini to the :{nexus}\conf directory:
* Be sure to remove the .in extension so that the file is named wrapper.conf.

### **Logs Directory**

* The Log directory was created above {nexus}\log\ directory. The default wrapper.conf files will place a wrapper.log files in this log directory.

### **Application Command Line: \_Profile Module**

* Before the Wrapper can be configured to launch the \_Nexus Application, you will need to know the full Java command that is normally used.
* Most applications make use of a batch file to build up the actual command line. \_Nexus Module is launched with a batch file: \_Nexus\_Run.bat (this is in the nexus\bin\ directory).
* If you run \_Nexus\_Run.bat you should get:



***Whenever a directory changes (say the JVM or the application directory) the \_Nexus\_Run.bat file must be changed to point to where the directory has moved. Otherwise the wrapper using the batch file will not work properly***.

### **Modifying the \_Nexus\_Run.bat file**

* Should a directory change or an upgrade occur (say with the JVM) the following lines may/will need modification in the Run1.bat file: **set DIRNAME= {nexus}\bin**
* If the path to the \_Nexus Module directory changes, this line must be adjusted to reflect the changes and point to the new directory:

**set RUNJAR=%DIRNAME%** **\_Nexus.jar**

* \_Nexus.jar is the executable Java Archive file (JAR) for the \_Nexus Module.
* If a new JAR file is created to replace the \_Nexus Jar file, then this line must reflect the new JAR file name: **set JAVA\_HOME=<drive>:\j2sdk1.4.2\_11**

This is the location of the JVM directory. If this directory is relocated or the JVM upgraded, then this line must be modified to reflect the new changes:

**ECHO CLASSPATH: "% \_Nexus\_CLASSPATH%" com.**<relevant\_path>**.nexus.\_NEXUS\_Thread.Main %ARGS%**

This line tells the wrapper which Class file within the com.cablebahamas..nexus package has the main class to launch the application. Should the source code change and another point of entry is chosen to launch the application, this line must change to point to the new Class file.

### **Modifying the warpper.conf files**

In order to be able to use the run1.batch (from {nexus}\bin) and the Java command lines with the Wrappers, we need to break up their components. The changes to address this issue were made in the wrapper.conf files. These changes are detailed below.

### **Java Executable**

* Within the wrapper.conf files in the {nexus}\conf directory is the following line: **wrapper.java.command=<drive>:\j2sdk1.4.2\_11\bin\java**
* This is the path to the Java interpreter. If the directory changes or the JVM is updated this line needs to be modified to reflect those changes.

### **Java Arguments**

* Most applications provide a number of parameters to the Java executable when it is launched. The NEXUSModule command lines will only have one such property:

**#wrapper.java.additional.1=-Dprogram.name =\_Nexus\_Run.bat**

### **Java Classpath**

* The classpaths are configured using the wrapper.java.classpath.<n> properties.
* The Wrapper requires that the classpaths be broken up into its individual elements.

* Because we will also be making use of the Wrappers, it is necessary to include the wrapper.jar files as well:

**For the NEXUS module:**

* wrapper.java.classpath.1={nexus}\lib\wrapper1.jar
* wrapper.java.classpath.2=<drive>:\j2sdk1.4.2\_11\lib\tools.jar
* wrapper.java.classpath.3={nexus}\bin\\_NEXUS.jar

***If the directory changes then these lines need to point to the new class paths to reflect those changes.***

### **Java Main Class**

* The final component of the command used to launch NEXUSModule is the main class;

**com.**<relevant\_path>**.nexus.\_NEXUS\_Thread**

* The main class executed by Java is specified by using the wrapper.java.mainclass property.
* Because the NEXUS Module main class does not know how to communicate with the Wrappers, we will set the main classes to be the full class name of WrapperSimpleApp.

**wrapper.java.mainclass=org.tanukisoftware.wrapper.WrapperSimpleApp**

* The Nexus Module main class is then specified as the first application parameter in its wrapper.conf file (see below):

**wrapper.app.parameter.1=**

**com.**<relevant\_path>**.nexus.\_NEXUS\_Thread**

### **Library Path**

* In order to use the Wrappers, the Library property has to be set.
* The Wrappers makes use of a native library to control interactions with the system.
* This file Wrapper.DLL needs to be specified on the library path supplied to the JVM.

For the Nexus Module: **wrapper.java.library.path.1={nexus}\lib**

### **Wrapper NT/2000/XP Service Properties**

* The final step is to set the Windows specific NT/2000/XP Service Properties

#### **For the Nexus Module (within its wrapper.conf file)**

# Name of the service

wrapper.ntservice.name=\_NEXUS Service

# Display name of the service

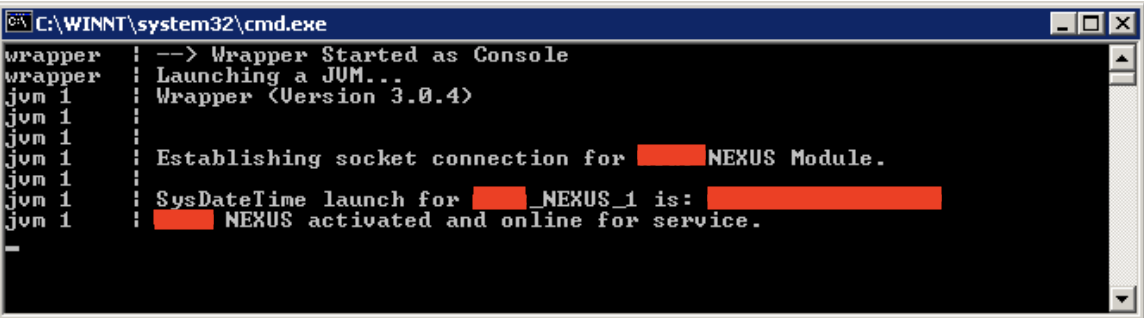
wrapper.ntservice.displayname= \_NEXUS Service

# Description of the service

wrapper.ntservice.description= \_NEXUS Service

### **Launching the Services**

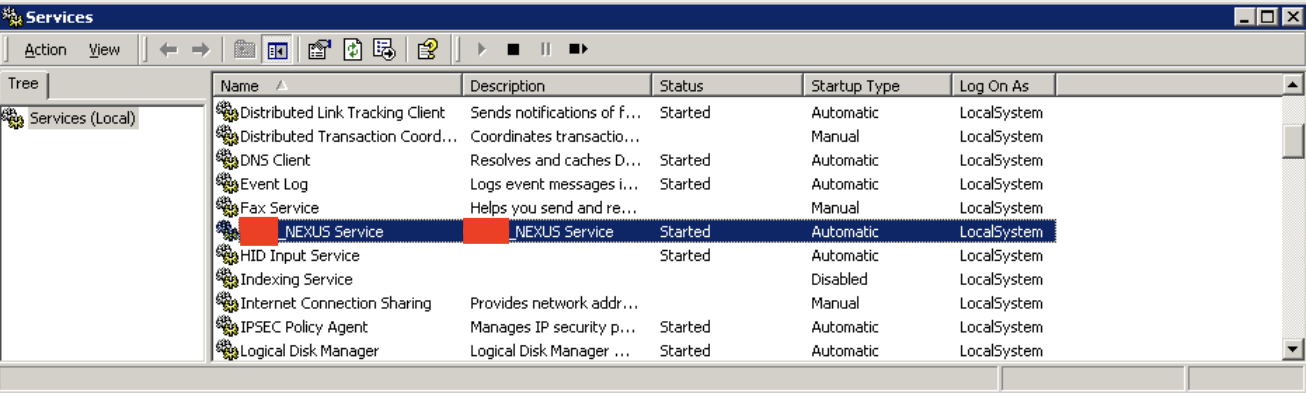
* Before launching the service, try running the application once to verify the configuration.
* To do this run: **{nexus}\bin\ \_NEXUS\_App.bat script.**
* A successful output should appear as:



* To run the NEXUS Module as services execute:

**{nexus}\bin\ Install\_NEXUS\_App-NT.bat script.**

* Verify that the service exists by entering the NT service utility to view the service module.



* To uninstall the services run:

**{nexus}\bin\Uninstall\_NEXUS\_App-NT.bat script**