

Java Contents

Java

Java jserver class

Java classpath environment variable

Java www.jsoftware.com applets

Java examples

Java sandbox and applet security

Java

J Automation objects JEXEServer and JDLLServer are available to Java programmers. A Java programmer can easily use the power of J in building applications and applets.

J Automation objects and the corresponding Java support are only available in the Windows 95 and NT versions of J.

J User Manual chapter OLE gives an overview of J Automation objects. The term Automation supersedes the term OLE.

Automation is a Microsoft technology and it is supported directly in the MS J++ Java Development system, the MS Java Virtual Machine (JVM) jview.exe, and in Internet Explorer.

There are two problems with this direct MS support of Automation. JVMs from other vendors (e.g., Sun's java.exe) and Netscape browsers can't run Java programs that use Automation objects. There are also serious limitations in mapping data between J and Java with the MS Automation support.

These problems were solved by building a Java class that wraps the J Automation objects in a way that is identically usable in all JVMs and browsers. This Java class is the jserver class and it is distributed with the J system.

The best way to learn how to use J in Java is to study the examples. Run them, build them in your Java development environment, and experiment. Stepping through with a debugger is a good way to study the examples.

Java jserver class

Java classpath environment variable

Java www.jsoftware.com applets

Java examples

Java sandbox and applet security

Java jserver class

The jserver class is a wrapper for the J Automation objects JEXEServer and JDLLServer. The jserver.class is in package isi.jserver (Iverson Software Inc.) and is distributed with the J system. To use a J object you import isi.jserver into a Java source file and use the class methods and fields.

The jserver class provides identical access for all JVMs to J Automation objects and it provides methods that simplify mapping J data to Java. The jserver class uses native methods in a C++ dll. There are two versions of the native method dll: jsnmms.dll (J Server Native Method MS) and jsnmns.dll (Netscape and other JVMs).

The jserver class uses native methods and must exist in the local file system. The class and native method dlls must be in the J system\extras\java\classes\isi\jserver directory. This path is added to the Java classpath environment variable when J is installed.

```
// constants - results of getType()
final static int TYPEBOOL=1;
final static int TYPEBYTE=2;
final static int TYPEINT=4;
final static int TYPEDOUBLE=8;
final static int TYPEBOX=32;

// methods
synchronized void start(int dllv) // 0 for JEXEServer, 1 for JDLLServer
int close() // close J object
int Do(String s)
int DoR(String s) // Do with formatted result
int Show(boolean b)
int Log(boolean b)

// methods for getting J data
int Get(String s) // get data for J variable

int getType()
int getRank()
int getElements()
int[] getShape()

boolean getBool()
char getChar()
int getInt()
double getDouble()

boolean[] getBools()
byte[] getBytes()
int[] getInts()
double[] getDoubles()
String getString()
```

```
// methods for setting J data  
int Set(String s, Object x)
```

```
int Set(String s, byte x)  
int Set(String s, int x)  
int Set(String s, double x)
```

Java classpath environment variable

A Java class is a .class file that contains the compiled result of java source files. When a JVM runs a program it dynamically loads the java classes required. When the jserver.class is required it must be loaded from your local file system. Different JVMs have different search paths for class files, but they all search the classpath path. Rather than add the jserver class file to different locations, the J installation updates the classpath environment variable to include the J system\extras\java\classes directory.

Java www.jsoftware.com applets

www.jsoftware.com has a few simple applets that you can run in your browser. When the applet uses the `jserver` class, it is loaded from the `J system\extras\java\classes` directory.

The `nyse` applet is different from the others in an important way because it reads data from a URL at the NYSE web site. Applets downloaded from the web run in the Java sandbox and are not allowed to read data from other web sites. The `nyse` applet would get a security violation if it were downloaded from the web site. The `nyse` applet class is in your `J` directory `system\extras\java\classes`. If this is in your classpath environment variable (as set by `J` setup) then the class is loaded from your hard drive and is not restricted to running in the sandbox. Installing applet classes on your hard drive and adding paths to your classpath have serious security implications. Be sure to read: [Java sandbox and applet security](#).

Java examples

The source for several Java programs is included in J directory system\examples\java. The examples are simple and could be understood just by reading, but it is helpful to use a debugging environment to step through the code to see exactly what happens.

Most of the examples are applets. Directory jstest contains an application. The jstest.class file is in the J system\extras\java\classes directory. You can run jstest with the MS JVM jview.exe or the Sun JVM java.exe. Bat file jv.bat runs jview.exe and ja.bat runs java.exe.

Java sandbox and applet security

A java applet downloaded from the web runs on your system in a secure environment called the sandbox. An applet running in the sandbox is not allowed to do anything that could violate the security of your system. In particular it can't read or write files. It is very important that a J object created in the sandbox obeys the rules. J foreigners (! :) and wd commands that could violate sandbox rules are disabled when the J object is created in the sandbox.

Important exception: loading ijs script files is allowed.

Be very careful about adding applet classes to your hard drive or changing your classpath. An applet loaded from your hard drive has complete access to your system. Allowing a strange applet to run from your hard drive is just as dangerous as running a strange exe program on your system.

It might sometimes be useful to explicitly request sandbox security completely independently from Java and Automation. For example, you might want sandbox security to run a script downloaded from the net. You trust the script, but setting sandbox security makes it safer. You can explicitly set sandbox security:

```
wd'security 1'      NB. wd sandbox security
9!:24 [ 1          NB. !: sandbox security
```