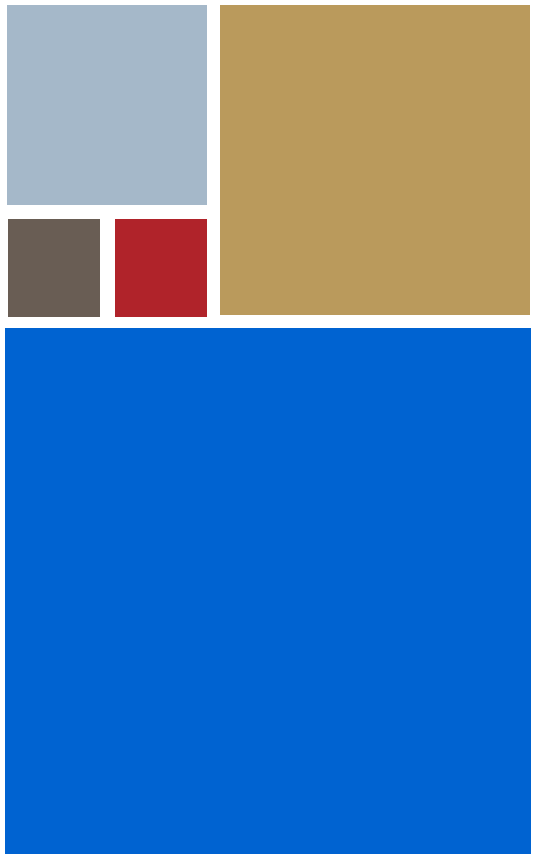


Home

Getting Started with PersonalJava™ Solution for OS-9® (PowerPC)

Version 3.1

RadiSys
THE POWER OF WE

www.radisys.com

Revision C • July 2006

Copyright and publication information

This manual reflects version 3.1 of PersonalJava™ Solution for OS-9.

Disclaimer

The information contained herein is believed to be accurate as of the date of publication. However, RadiSys Corporation will not be liable for any damages including indirect or consequential, from use of the OS-9 operating system, Microware-provided software, or reliance on the accuracy of this documentation. The information contained herein is subject to change without notice.

Reproduction notice

July 2006

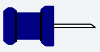
Table of Contents

Chapter 1: Introduction

This manual provides you with the information you need to get started with PersonalJava™ Solution for OS-9®.

For More Information

Refer to the current version of OS-9 Release NOtes for possible last-minute updates to PersonalJava™ Solution for OS-9 or the PowerPC board.



Note

Before proceeding, be certain you have installed either OS-9 for Embedded Systems or the OS-9 Board Level Solution (BLS) for your processor, on your Windows-based host system. If you do not have either of these packages, contact your OS-9 supplier.

For More Information

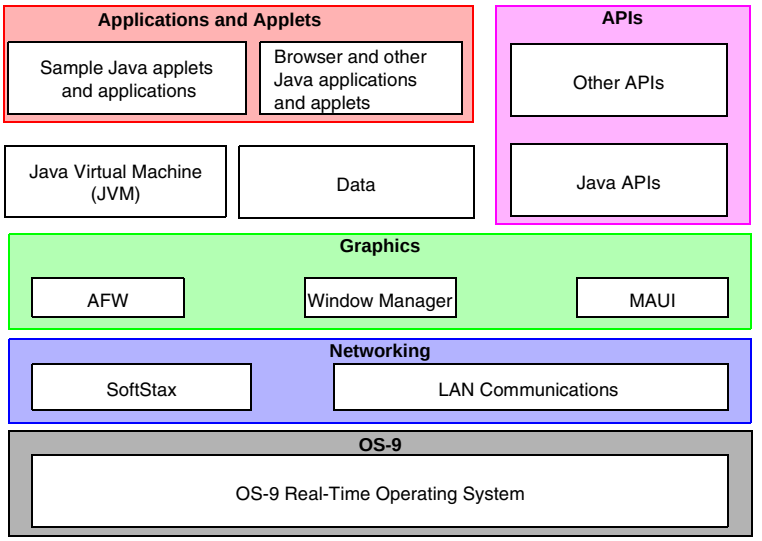
Refer to the CD-ROM insert for information about installing PersonalJava™ Solution for OS-9 on your Windows-based host platform.

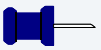
PersonalJava™ Solution for OS-9 Runtime Components

PersonalJava™ Solution for OS-9 is a complete system software solution for developing Java-enabled devices. The PersonalJava™ Solution for OS-9 system consists of a scalable real-time operating system with specific software modules that help you create Java enabled devices without worrying about system software customization.

Figure 1-1 shows the PersonalJava™ Solution for OS-9 architecture. Each software subsystem found in PersonalJava™ Solution for OS-9 is defined in the following sections.

Figure 1-1 PersonalJava™ Solution for OS-9 Runtime Components





Note

Many of these components were installed with your **OS-9 for Embedded Systems** or **OS-9 Board Level Solution** package. You must have installed one of these packages prior to installing PersonalJava™ Solution for OS-9. Contact your OS-9 provider for a copy of one of these packages.

OS-9

At the core of PersonalJava™ Solution for OS-9 is OS-9 and its support modules.

OS-9 is an architecturally advanced, high performance real-time operating system available for several microprocessor families. At its core is the OS-9 stand-alone microkernel.

Coupled with the power of the microkernel, the unique modular architecture of OS-9 enables dynamic loading of any OS-9 system or user application module while the system is up and running.

For More Information

Refer to the OS-9 manual set for more information about the operating system.

Networking

The ability to communicate with other computers or devices is essential for a Java-enabled device. PersonalJava™ Solution for OS-9 uses the standard SoftStax® I/O implementation so a variety of transport layers can be used with Java.

SoftStax

SoftStax provides a consistent application-level interface using a variety of networking protocols. The protocols necessary for using PersonalJava™ Solution for OS-9 are included in LAN Communications.

LAN Communications

The Microware LAN Communications software consists of a TCP/IP protocol stack with UDP support, SLIP/CSLIP support, PPP support, and drivers for supported hardware.

For More Information

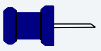
Refer to the SoftStax and LAN Communications manual sets for more information about this implementation.

Graphics

One of the strengths of Java as a programming language is its support for graphics. To handle graphics, PersonalJava™ Solution for OS-9 uses four components: MAUI®, Window Manager, the Application Framework (AFW), and the Java Abstract Windowing Toolkit (AWT).

Multimedia Application User Interface (MAUI)

MAUI is a high-level library that manages the display of graphics, text, and user input as well as audio.



Note

Audio is available only on targets supporting sound devices.

For More Information

Refer to the MAUI manual set for more information about the MAUI system.

Window Manager

The PersonalJava™ Solution for OS-9 Window Manager is a MAUI application that manages windows. Three versions of the Window Manager are available, each with different levels of functionality.

For More Information

Refer to ***Using PersonalJava™ Solution for OS-9*** for more information about the Window Manager.

Application Framework

The AFW is a class library that contains the code necessary to display GUI components and handle events for an interactive application.

Java Abstract Windowing Toolkit

The PersonalJava™ (PJAVA) environment includes an AWT package that allows Java applications to display GUI components, render images, draw graphics primitives, and respond to events. This package is standard across all PJAVA implementations, although some features are optional in PersonalJava implementations. All optional AWT functionality is fully supported in Microware's PersonalJava™ Solution for OS-9.

Java Virtual Machine

Consumer devices that interpret Java applications must contain the JVM. Java applications are comprised of Java classes consisting of byte codes.

Java byte codes are machine-independent and interpreted by the JVM. The purpose of the JVM is to interpret these Java byte codes and initiate appropriate actions on the host platform. In addition to executing byte codes in all classes within the system, the JVM also handles signals and Java exceptions, manages RAM, and is responsible for the simultaneous execution of multiple threads within the context of the JVM process.

Applications and Applets

Along with the basic system components, Microware has included several sample applications and applets on the PersonalJava™ Solution for OS-9 CD.

Sample Applications

Several sample applets have been included in this package. They are located in `MWOS\SRC\PJAVA\DEMO`. Additional sample applets from Sun are located in `MWOS\DOS\jdk1.1.8\demo`.

Additional Java Tools

The following describes JavaCodeCompact.

Running Java on a Diskless System

PersonalJava™ Solution for OS-9 includes a tool called JavaCodeCompact that enables sets of Java class files to be pre-loaded in RAM or placed in ROM. This is accomplished by pre-processing the class files into an assembly language file that is eventually turned into a module. The module can then be loaded at run-time or loaded into the ROM of the device. This process eliminates the need to have the class files themselves, often times called `classes.zip`, resident on the device.

For More Information

Refer to ***Using JavaCodeCompact for OS-9*** for instructions on using this tool in the OS-9 environment and refer to ***Using PersonalJava™ Solution for OS-9*** for information about creating Java applications for a diskless OS-9 target.

Java Development Tools

Due to the portable nature of Java, users can develop their Java applications using any of the GUI-based Java development packages on the market. Some of these include Metrowerks CodeWarrior, Sunsoft's Java Workshop, and Symantec's Visual Cafe. As long as the output of the development environment is standard Java class files containing standard byte codes, the code is compatible with PersonalJava™ Solution for OS-9.

Standard Java class files contain a great deal of information about the source code from which they were compiled, including symbol names. With the appropriate tools, it is possible to de-compile Java code into an almost exact replica of the source code. Some of these tools address this problem by munging or obfuscating the object code so de-compilation is not as easy. Refer to the numerous Java-related web sites and UseNet news groups for information on these tools.

For More Information

For more information about CodeWarrior, visit the Metrowerks website at `http://www.metrowerks.com/`.

For more information about Java Workshop, visit the Sun website at `http://www.sun.com/`.

For more information about Visual Cafe, visit the Symantec website at `http://www.Symantec.com/`.

Windows® Java Development Kit (JDK)

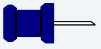
To make it easier for you to perform native method work, Microware has included the Windows JDK v1.1.8 in the package on the host system.

The `javah.exe` executable on the host machine has been modified to generate code that works with the Microware UltraC/C++ compiler.

The pre-loader classes are contained in the `jcc.zip` file. This file is on the Windows host machine in the `\MWOS\DOS\jdk1.1.8\lib` directory.

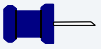
Chapter 2: Getting Started on a Disk-based Target

This chapter explains how to install PersonalJava Solution for OS-9 on a disk-based target and how to run the PersonalJava Solution for OS-9 examples.



Note

The following procedures assume that your target system is disk-based. Refer to ***Using PersonalJava Solution for OS-9*** for information about using a diskless target system.



Note

You must install Microware OS-9 for PowerPC before installing PersonalJava Solution for OS-9.

The procedures in this chapter use the `D:\` drive on your host (this may vary depending on where you chose to install your PersonalJava Solution for OS-9 package).

This chapter describes using PersonalJava Solution for OS-9 on the Motorola PowerPC-based MTX604. The same procedures can be applied to other disk-based targets. You can modify the procedures as necessary for your particular target.

System Requirements

Note

The hardware and software requirements for using PersonalJava Solution for OS-9 on your host and target are listed in the appropriate ***OS-9 Board Guide***.

Installing PersonalJava Solution for OS-9 on Your Target

Before you begin, boot up your target and establish a virtual terminal connection between the PC and the target. This connection is described as the target console throughout the remainder of this document. If you are using a physical terminal as the target console instead of a virtual terminal, installation instructions remain identical.

Installation of Java on the target is initiated on the PC and finished on the target. Therefore, the following instructions are divided into these stages:

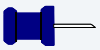
- **Stage 1: Configuring Your Boot**
- **Stage 2: On the PC**
- **Stage 3: On the Target**

Stage 1: Configuring Your Boot

These steps should guide you through the process of using the Configuration Wizard to build a boot image for a target that is Java ready. Java ready is defined by the following guidelines:

- MAUI and SoftStax components are included in the boot
- a network connection is available (through SoftStax)
- mshell is used

The process that follows assumes that you have already been through the Getting Started manual for your particular board and you have the means to use a different high-level boot.



Note

The Java Development Kit (JDK) as shipped from Sun is targeted strictly at desktop environments. PersonalJava Solution for OS-9 may be used on either disk-based or diskless systems. The examples you will be running require a system disk device for a file system.

For More Information

If you have not set up a system disk device on your target, refer to the ***OS-9 for MTX Board Guide*** for information regarding this procedure.

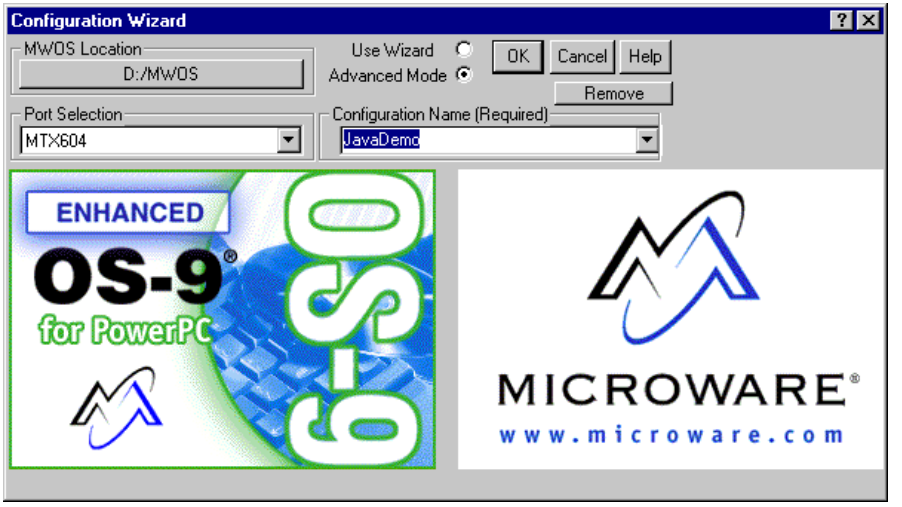
Choose the appropriate board guide from the list of documents if you are using a target system other than MTX.

Configuring your boot involves the following tasks:

- **Task 1: Creating the Boot Image**
- **Task 2: Transferring the high-level OS-9 Bootfile the Target Board**
- **Task 4: Checking the Boot**
- **Task 5: Setting Up the System Disk**
- **Task 6: Transferring the OS-9 Utilities to the Target OS-9 System (Optional)**

Task 1: Creating the Boot Image

Step 1. From the Windows Start Menu, select `Programs -> <your OS-9 package> -> Microware Configuration Wizard`. The following window appears:



Complete the following on the opening screen:

- Fill in the `MWOS Location` field with the location of the MWOS tree installed when you installed your OS-9 for Embedded Systems or OS-9 Board Level Solution.

  Example: `D:\MWOS`

- Select your target's board model in the `Port Selection` box.
- Select the `Advanced Mode` option.
- Fill in the `Configuration Name` field with the name of the Java demo configuration file `JavaDemo`. This file has been installed into the MWOS tree on your PC at `D:\MWOS\OS9000\<port proc>\PORTS\<portname>\BOOTS\INSTALL\<boardname>\JavaDemo.ini`

Note

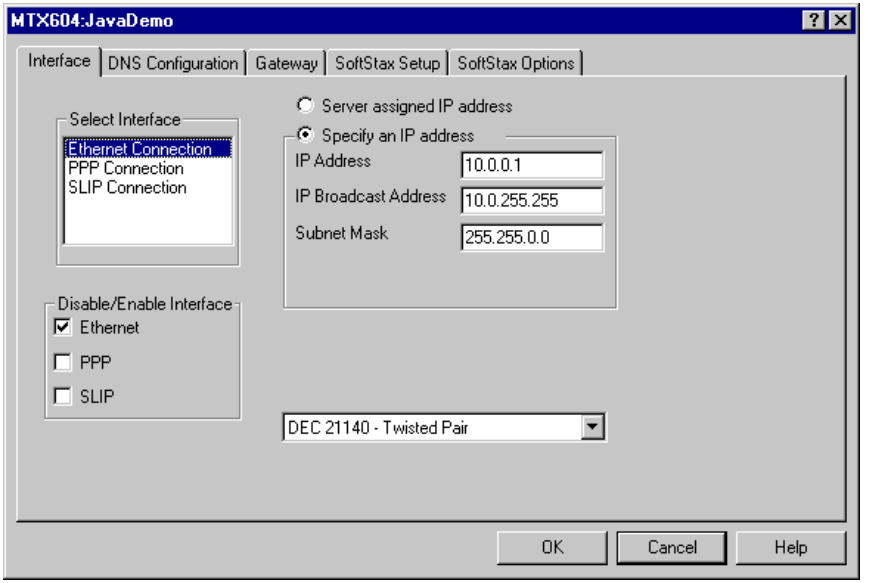
On subsequent runs, the Wizard will automatically prepends the board model to the configuration name. There is no need to type this portion of the name.

- Click on the `OK` button.

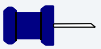
For More Information

Refer to the online help available in Wizard for additional information about using the wizard.

Step 2. From the Wizard menu options, select `Configure -> Bootfile -> Network Configuration`. Ensure that Ethernet Connection is selected in the list and that the checkbox is selected next to Ethernet:



Fill in the ethernet setup fields with the ethernet information for your target:

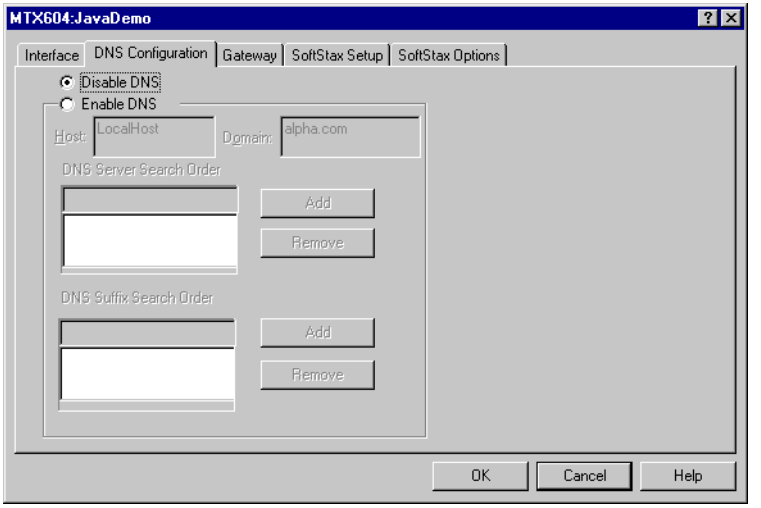


Note

You may need to contact your network administrator for the following information.

- `IP Address` - type your IP address in this field
- `IP Broadcast` - type your IP Broadcast in this field
- `Subnet Mask` - type your Subnet Mask in this field

Step 3. Select the `DNS Configuration` tab at the top of the window. The following window appears:

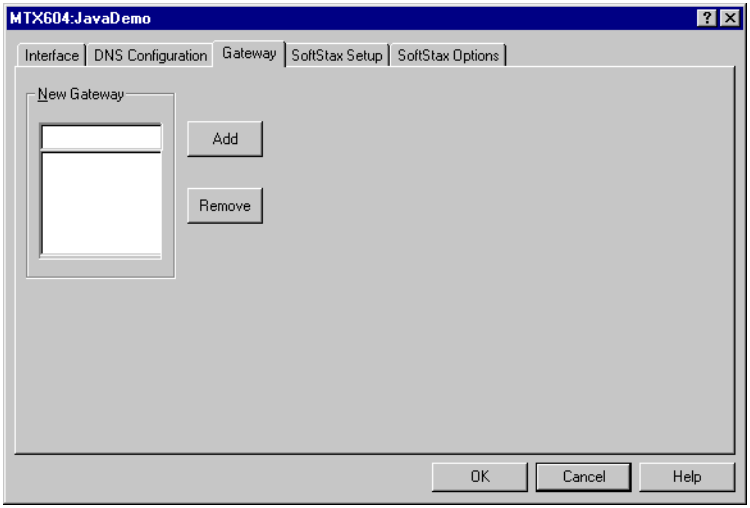


Fill in the following fields with the information for your target:

- Select `Enable DNS`.
- Enter your target's host name in the `Host` field.
- Enter your target's domain name in the `Domain` field.

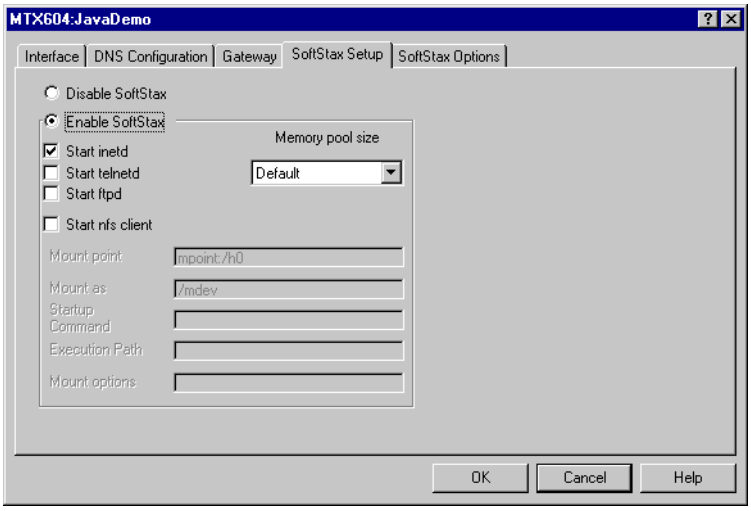
- Enter your target's DNS Server Search Order addresses. Add as many addresses as are valid for your location.
- Enter your target's DNS Suffix Search Order. Add as many suffixes as are valid for your location.

Step 4. Select the `Gateway` tab at the top of the window. The following window appears:



Enter your New Gateway network address. Add as many addresses as are valid for your location.

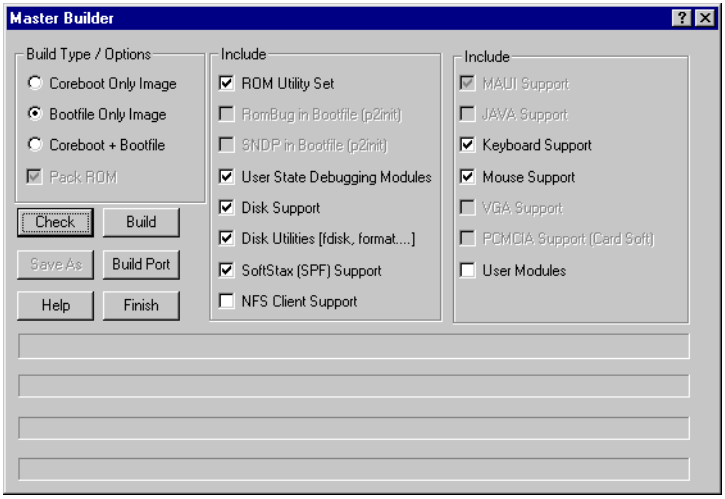
Step 5. Select the `SoftStax Setup` tab at the top of the window. The following window appears:



Fill in the following fields with your information:

- select `Enable SoftStax`
- select the `Start inetd` checkbox
- click `OK` at the bottom of the screen to close the window

Step 6. Build the Java Demo bootfile image. From the mwWizard menu options, select `Configure -> Build Image`. The following window appears:



Step 7. In the Master Builder window, click the `Check` button. Confirm that no errors are reported. Refer to Troubleshooting in the Wizard helpfile for more information about errors you may receive.

Step 8. In the Master Builder window, click the `Build` button. The file `D:\MWOS\OS9000\<port proc>\PORTS\<portname>\BOOTS\INSTALL\PORTBOOT\bootfile` is created.

Step 9. Click `Finish`.

Step 10. Exit Wizard.

Task 2: Transferring the high-level OS-9 Bootfile the Target Board

Use whatever means is appropriate for transferring the bootfile to the target. For a disk-based MTX board it is simply FTPing the high-level boot to the root of the device /h0 and executing the command line:

```
bootgen /hs01fmt -q bootfile
```

Task 4: Checking the Boot

When the target auto boots you see OS-9 booting information displayed. The auto boot ends with the OS-9 prompt displayed. Complete the following steps to check the boot:

Step 1. Make sure the following components are burned into ROM and are present in the module directory. At the OS-9 prompt type the following command line:

```
mdir -e gfx "gx_*" cdb kx0 m0 maui maui_inp \
    maui_win mfm mp_msptr mp_xtkbd sc8042k
```

A module directory similar to the following is displayed:

```
            Current Module Directory

  Addr     Size      Owner     Perm Type Revs  Ed #  Lnk  Module name
-------- -------- ----------- ---- ---- ---- ----- ----- --------------
0032e150      304      0.0     0555 5    8000   200     1 cdb
0032e280      568      1.0     0555 Desc 8000   300     1 gfx
0032e4b8    34344      0.0     0555 Driv a000     1     1 gx_cl5446
00230b58     2016      0.0     0555 Desc 8000     1     1 kx0
00336ae0   251824      1.0     0555 Subr 8000   305     1 maui
0030d4b0    21728      0.0     0555 Prog 8001   212     1 maui_inp
00312990    74536      0.0     0555 Prog 8001   219     1 maui_win
0030cbf8     2232      0.0     0555 Fman a000    14     1 mfm
00329930     5088      0.0     0555 Subr 8001   212     1 mp_msptr
0032ad10     9064      0.0     0555 Subr 8001   212     1 mp_xtkbd
0022cf50    13360      0.0     0555 Driv a000    44     1 sc8042k
```

Step 2. Make sure your network is running and ready for a connection. At the OS-9 prompt type:

```
procs -e
```

This allows you to view the processes running on the system.

An entry, similar to the following, is displayed in the active process list:

```
Id PId Grp.Usr  Prior  MemSiz Sig S   CPU Time   Age Module & I/O
  3   0   0.0     128  100.00k   0 s       0.07  0:12 inetd <>>>nil
```

The `inetd` module indicates the network is running and ready for a connection.

If it is not running, type the following at the OS-9 prompt:

```
inetd<>>>/nil&
```

Task 5: Setting Up the System Disk

Refer to ***OS-9 for MTX Board Guide*** for the procedure for setting up your system disk.

Choose the appropriate board guide from the list of documents if you are using a target system other than MTX.

Task 6: Transferring the OS-9 Utilities to the Target OS-9 System (Optional)

Once the system disk has been formatted, you may find it useful to copy the OS-9 utilities from the Windows host machine to your target system. The easiest method to accomplish this is to transfer the utilities from the Windows host machine to the OS-9 target using FTP. The following steps copy all of the OS-9 utilities from `D:\MWOS\OS9000\PPC\CMDS` on the Windows host machine to `/h0/CMDS` on the OS-9 target machine:

Step 1. Create the `CMDS` directory on the OS-9 target machine by typing the following in the FTP window:

```
makdir /h0/CMDS
```

Step 2. Choose `Start->Run` on the Windows desktop.

Step 3. Type the following in the Run dialog box and click the `OK` button:

```
ftp <target machine name>
```

Step 4. Log on to the OS-9 machine by typing the user name and password in the FTP (MS-DOS Shell) window. The default user name and password for OS-9 machines is `super` and `user`.

Step 5. Change to the OS-9 Utilities directory on the Windows host machine by typing the following in the FTP window:

```
lcd D:\MWOS\OS9000\PPC\CMDS
```

Step 6. Change to the `CMDS` directory on the OS-9 target by typing the following in the FTP window:

```
cd /h0/CMDS
```

Step 7. Change to binary transfer by typing the following in the FTP window:

```
bin
```

Step 8. Transfer the utilities files to the target by typing the following in the FTP window:

```
prompt
mput *
quit
```

Step 9. Set the file attributes for the utilities on the OS-9 target machine by typing the following in the console window:

```
chd /h0/CMDS
attr -peprgegrewr *
```

Stage 2: On the PC

Complete the following steps on the host PC:

Step 1. Choose `Start->Run` on the Windows desktop.

Step 2. Type the following in the Run dialog box:

```
ftp <target machine name>
```

Step 3. Click the `OK`.

Step 4. Log on to the OS-9 machine by typing the user name and password in the FTP (MS-DOS Shell) window. The default user name and password for OS-9 machines is `super` and `user`.

Step 5. From the host PC, navigate to the directory where the PersonalJava example modules are installed by entering the following in the FTP window:

```
lcd D:\MWOS\OS9000\<port proc>\PORTS\<port name>\CMDS\BOOTOBJS\PJAVA
```

Step 6. Create a PJAVA directory on the root of the hard disk:

```
mkdir /h0/PJAVA
```

Step 7. Change to the PJAVA directory on the target system by entering the following command in the FTP window:

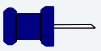
```
cd /h0/PJAVA
```

Step 8. Transfer files by typing the following command in the FTP window:

```
bin
prompt
mput pjruntime *.mod
```

Step 9. Type the following command in the FTP window:

```
ascii
put go.demo
```



Note

Due to the variety of FTP applications available for Windows 95 and Windows NT, Microware is unable to accurately describe how to accomplish steps 1 through 4. Consult your FTP documentation for an exact description.

Stage 3: On the Target

Using mshell, initiate the following command to start the installation process:

```
$ profile /h0/PJAVA/go.demo
```

The following should display on the screen:

```
Loading PersonalJava run-time components...
Loading PersonalJava for OS-9 examples...
+6
+7
Starting PersonalJava
+9
```

On the OS-9 target machine display, you will see the PersonalJava Demo.

The `go.demo` mshell script sets several system parameters to prepare the OS-9 target for running PersonalJava Solution. These system parameters include:

- setting the `CLASSPATH` environment variable
- setting the `JAVA_HOME` environment variable
- loading the file `/h0/pjruntime`
- initializing the devices kx0, m0, and mm
- starting the MAUI Input process
- starting the PersonalJava Solution for OS-9 Window Manager
- starting the PersonalJava Virtual Machine

Refer to the mwWizard help file for an in-depth discussion of setting system parameters in the `init` module.

Complete the following steps on the target:

Step 1. Type the following at the OS-9 prompt to see which processes are running on the OS-9 target:

```
procs -e
```

A process list similar to the following is displayed:

```
 Id PId Grp.Usr  Prior  MemSiz Sig S    CPU Time   Age Module & I/O
  2   0   0.0     128   56.00k   0 w        0.10  1:34 mshell <>>>term
  3   2   0.0     128   76.00k   0 s        0.01  1:35 inetd <>>>nil
  4   0   0.0     128   20.00k   0 e        2.63   ??? spf_rx
  6   2   0.0     200   12.00k   0 s        0.01  1:34 maui_inp <>>>nil
  7   2   0.0     200  160.00k   0 s        0.47  1:34 winmgr <>>>nil
  8   7   0.0     256  156.00k   0 s        9.09  1:34 maui_win <>>>nil
  9   2   0.0     128 6020.00k   0 s     7:53.87  1:34 pjava <>>>nil
 10   2   0.0     128   44.00k   0 *        0.02  0:00 procs <>>>term
```

Step 2. Type the following at the OS-9 prompt to see which environment variables have been set:

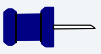
```
printenv
```

A list similar to the following is displayed:

```
SHELL=mshell
CLASSPATH=/mm/lib/classes.zip:.:/mm/GasPump.jar:/mm/PJDemo.jar:/mm/LaunchPad.jar
JAVA_HOME=/mm
_SHELLPARAMS=ne,no,f,nd,ns,b,ni,nr,nm,nv,q,nh
PWD=/r0
```

The variables listed include the following:

- the `CLASSPATH` variable. This variable specifies the location of the `.jar` files used as Java libraries.



Note

You can append additional paths to this path if you have created your own `.jar` files.

- the `JAVA_HOME` variable. This variable specifies the location of the Java properties files.