

INSTALLING THE TOTALVIEW JNIBRIDGE



VERSION 3.0



Copyright © 2007, 2008 by TotalView Technologies. All rights reserved

Copyright © 1998–2007 by Etnus LLC. All rights reserved.

Copyright © 1996–1998 by Dolphin Interconnect Solutions, Inc.

Copyright © 1993–1996 by BBN Systems and Technologies, a division of BBN Corporation.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise without the prior written permission of TotalView Technologies.

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

TotalView Technologies has prepared this manual for the exclusive use of its customers, personnel, and licensees. The information in this manual is subject to change without notice, and should not be construed as a commitment by TotalView Technologies. TotalView Technologies assumes no responsibility for any errors that appear in this document.

TotalView and TotalView Technologies are registered trademarks of TotalView Technologies. TVD is a trademark of TotalView Technologies.

TotalView Debugger uses a modified version of the Microline widget library. Under the terms of its license, you are entitled to use these modifications. The source code is available at <http://www.totalviewtech.com/Products/TotalView/developers>.

All other brand names are the trademarks of their respective holders.

Contents



Installing the JNIBridge

Introduction	1
Installation	1
Installing the TotalView Debugger	1
Installing the JNIBridge Plugin	2
Build the JNI Example Program	2
JNI Debugging Using Eclipse	3
Start Eclipse	3
Setup JNIBridge Preferences	3
Set Breakpoints	3
Create JNIBridge Debug Configuration	3
Start Debugging	4
Known Issues	4

Installing the JNIBridge



Introduction

This guide provides instructions for installing the JNIBridge. It also steps you through a small "hello world" JNI example. This example shows demonstrates how the JNIBridge lets you "seamlessly" follow code execution from Java into C/C++.

The following are JNIBridge requirements:

- Linux-x86 (preferably Red Hat ES 4)
- Eclipse 3.2.2 or Eclipse 3.3
- Java 1.5
- TotalView Debugger 8.4 or later

Installation

Before you install the JNIBridge, TotalView Debugger (TVD), version 4.0 must already be installed. After it is installed, you can then install the JNIBridge plugin for Eclipse.

Installing the TotalView Debugger

TotalView Debugger, version 4, is available on our web site. You can download a copy by going to <http://www.totalviewtech.com/eval/formTV.htm>. If you do not have a license to run TVD, please fill in our evaluation form and we will quickly send you one.

Installation instructions for the TVD are available at http://www.totalviewtech.com/Documentation/latest/pdf/install_guide.pdf.

Installing the JNIBridge Plugin

You can download TotalView JNIBridge 3.0 by going to <http://www.totalviewtech.com/download.htm>. The tar file that you will download contains our Eclipse plugin.

Before installing the JNIBridge Eclipse plugin, make sure you have the Eclipse 3.2.2 or 3.3 SDK installed and working. You can download those SDKs for linux-x86 from <http://www.eclipse.org/downloads/>.

While Eclipse provides many ways to install software, we recommend that you install the JNIBridge for Eclipse 3.2 in the following way:

- 1 Copy the **com.totalviewtech.jnibrige_3.0.0-0-A-3.tar** file to the Eclipse 3.2.2 plugin directory. Normally, this is `[ECLIPSE-INSTALL-DIR]/plugins`.
- 2 Untar the file using the following command:

```
tar xf com.totalviewtech.jnibrige_3.0.0-0-A-3.tar
```

This will create a `com.totalviewtech.jnibrige_3.0.0-0-A-3` directory under `plugins`.
- 3 Remove the tar file.

```
rm com.totalviewtech.jnibrige_3.0.0-0-A-3.tar
```

Build the JNI Example Program

We have provided a simple JNI example that demonstrates debugging across the native interface. You can download the examples tar file from our web site.

- 1 Copy the **hello-world-example.tar** file to any work space directory and untar the file.
- 2 The tar file contains a **readme.txt** with instructions on building the example. You should try building and running the example outside of Eclipse. This lets you know that your paths are correct.

JNI Debugging Using Eclipse

This section describes how to setup and use the JNIBridge to debug JNI code.

Start Eclipse

Start Eclipse and create an Eclipse project for the JNI example. After your project is set up, run the example without debugging shut off. This makes sure your project paths are correct.

Setup JNIBridge Preferences

You need to setup a JNIBridge preference that tells the JNIBridge where you installed the TotalView debugger.

- 1 Select the Eclipse **Windows > Preferences** menu item.
- 2 Select the **JNIBridge** topic in the left column.
- 3 Enter the installation path for TotalView 8.4 and press **Ok**. For example, if this path were `/usr/toolworks`, you would type `/usr/toolworks/totalview.8.4.0-0/bin`

Set Breakpoints

Stopping inside a JNI method call means that you need to set a breakpoint on the native method declarations. *Setting this breakpoint is the only requirement.* In this **HelloWorld** example program, the `helloThere()` native method is declared on line 28.

```
public class HelloWorld {
    ...
    // Native method declaration
    // Place breakpoint on this line
    public native void helloThere();
    ...
}
```

You may also want to set a breakpoint at the beginning and end of the `main()` method as this will help you follow process execution.

Create JNIBridge Debug Configuration

Eclipse comes with predefined debug configurations and the JNIBridge plugin adds another one for debugging JNI code.

- 1 Select the **Run > Debug** menu command (in Eclipse 3.3 the menu item is called **Run > Open Debug Dialog**) or select **Debug** from the drop down menu on the debug button in the toolbar (in Eclipse 3.3 the drop down item is called **Open Debug Dialog**).

- 2 From the displayed **Debug Configuration** dialog box, select the **JNIBridge** option on the left and then press the **New configuration** button.
- 3 Use the newly displayed JNIBridge debug configuration dialog box to name this configuration. You can click **Stop in Main** to have greater control of the process if you did not already add the breakpoint in **main()**. Finally, press the **Close** button.

Start Debugging

Start your debug session by selecting the JNIBridge debug configuration you created in the previous step. The Java debugger should be stopped at the breakpoint at the beginning of the **main()** method. You can now step through your code to the call to **helloThere()**, which is our native method. Step over this call.

The TotalView Process window will be stopped at the beginning of the native method implementation. You are now ready to explore your native code through TotalView as you control its execution. When you are done, press **Go** in the TotalView Process window's Toolbar. Execution control returns to the Eclipse Java debugger.



*As you step through your Java code, you may end up stopped at the breakpoint on your native method or in Java's implementation of **ClassLoader.findNative()**. Rather than try to Step Into all the core Java calls, you should Step Out or Over it. TotalView will then stop execution in the native method.*

Known Issues

- 1 If you exit Eclipse while a JNIBridge debug session is running, it may leave TotalView running. You should manually exit out of TotalView.
- 2 The JNIBridge only supports one debug session at a time.
- 3 If you change the JNIBridge preferences after the JNIBridge is launched, you must restart Eclipse for those changes to take effect.
- 4 If you try to run a JNIBridge debug configuration and it fails, you must restart Eclipse to try again.
- 5 TotalView may pick up a **segv** in the Java Virtual Machine, if this happens press **Go** in TotalView. The JVM should continue normally.