Sun Cobalt™ Java™ Developer Kit (v1.3)

User Manual

Part Number: **070-00346-01**
Date: **05-2001**

Sun Microsystems, Inc.
555 Ellis Street
Mountain View, CA 94043
www.sun.com
www.cobalt.com

In the U.S.A.:

Phone	(888) 70-COBALT
	(650) 623-2500
Fax	(650) 623-2501

Outside the U.S.A.:

Phone	+1 (650) 623-2500
Fax	+1 (650) 623-2501

Contents

1 Sun Cobalt™ Java™ Developer Kit (v1.3) 1

Introduction 1

Installation 1

Important information for the Sun Cobalt Qube 3 appliance and Sun Cobalt RaQ XTR server appliance 2

JSP and servlet operation 3

Deploying JSPs and servlets on the Sun Cobalt Qube 3 appliance 4

Deploying JSPs and servlets on a Sun Cobalt RaQ server appliance 5

Web archive (.war) files 6

Creating a .war file 6

Deploying a .war file 7

Restarting the Web service 8

Compiling a Java™ servlet source file 9

Connecting to InterBase 11

Turn on the InterBase server 12

Assign a password to access the InterBase database server 13

Create a database 13

Customizing Apache Tomcat 15

Security Policy 15

Class path for database drivers and other classes 16

Disabling the Tomcat engine 17

Sun Cobalt™ Java™ Developer Kit (v1.3)

Introduction

The Sun Cobalt™ Java™ Developer Kit is a set of tools designed to help developers create Java-centric applications on the Sun Cobalt appliance platform. The kit includes the Java 2 Standard Edition (J2SE) development environment and the Apache Tomcat JavaServer Pages (JSP) / servlet engine.

The Sun Cobalt Java Developer Kit supports the following platforms: the Sun Cobalt RaQ™ 3 server appliance, the Sun Cobalt RaQ™ 4 server appliance, the Sun Cobalt RaQ™ XTR server appliance and the Sun Cobalt Qube 3 appliance.

Installation is a click away via the server's Web administration interface. For server appliances that ship with the InterBase® database, the InterClient™ Java database connectivity (JDBC) database driver has also been included in the kit.

Installation

The Sun Cobalt Java Developer Kit is distributed as a package file (.pkg file) that is uploaded to the Sun Cobalt server appliance.

- For a Sun Cobalt RaQ XTR server appliance or a Sun Cobalt Qube 3 appliance, load the package through the BlueLinQ tab.
- For a Sun Cobalt RaQ 3 and RaQ 4 server appliance, load the package through the **Install Software** screen.

Note: Only the RaQ Administrator can access these screens.

To access the **Install Software** screen, click **Maintenance** on the left and then click **Install Software** at the top.

The software for the Sun Cobalt Java Developer Kit requires a minimum of 128 MB of RAM on the server appliance. Table 1 shows the patches required for each platform to run the Sun Cobalt Java Developer Kit software.

Table 1. Required patches

Platform	Patch required
Sun Cobalt Qube 3 appliance	OS Update 1.0
Sun Cobalt RaQ 3 server appliance	OS Update 4.0
Sun Cobalt RaQ 4 server appliance	OS Update 1.0
Sun Cobalt RaQ XTR server appliance	No OS update required.

Important information for the Sun Cobalt Qube 3 appliance and Sun Cobalt RaQ XTR server appliance

On a Sun Cobalt Qube 3 appliance or a Sun Cobalt RaQ XTR server appliance, you cannot upload a .pkg file larger than 8 MB. The Sun Cobalt Java Developer Kit is larger than this limit.

A solution to this limitation is to load the file in the Web root of the server appliance and have the server appliance download the .pkg file from itself.

1. Download the .pkg file to your local machine from either the Sun Cobalt Web site (www.cobalt.com) or from the compact disc (CD).
2. Use a file-transfer-protocol (FTP) client to transfer files to the Sun Cobalt RaQ XTR server appliance or the Sun Cobalt Qube 3 appliance.
3. Log in to the server appliance as the user *admin*.
4. Put the .pkg file in the location appropriate to your server appliance:
    - On a Sun Cobalt RaQ XTR server appliance, cd to the directory `/home/sites/home/web`
    - On a Sun Cobalt Qube 3 appliance, cd to the directory `/home/groups/home/web`

5. Through a Web browser, access the Server Desktop user interface (UI) of the server appliance.
6. Click the **BlueLinQ** tab at the top.
7. Click **New Software** on the left. The “Available New Software” table appears.
8. Click **Install Manually**. The “Install Manually” table appears.
9. Click the URL radio button.
10. In the URL field, enter http://<name_of_local_host>/[name_of_the_.pkg_file].
11. Click **Prepare** to start the installation.

JSP and servlet operation

Once the Sun Cobalt Java Developer Kit is installed on the Sun Cobalt server appliance, the JSP and servlet functionality is enabled on the Web server. JSP and servlets are executed by the Tomcat engine. Tomcat is a plug-in to the Web server already installed on the server appliance.

Once you install the Sun Cobalt Java Developer Kit, it instructs the Web server to send certain requests to Tomcat for processing. Tomcat requires an execution environment known as a *context* to be set up before it can execute JSPs and servlets properly. A context is basically a directory structure on the server from which Tomcat can execute servlets and JSPs.

The Sun Cobalt Java Developer Kit automatically sets contexts for you. The kit contains a default context; you can also add new contexts by loading Java Web archive files (having the extension .war) into Web-content directories.

Note: To disable the JSP and servlet functionality, see “Disabling the Tomcat engine” on page 17.

Deploying JSPs and servlets on the Sun Cobalt Qube 3 appliance

On the Sun Cobalt Qube 3 appliance, the default context for Tomcat is the `/home/groups/home/web/` directory. When you install the Sun Cobalt Java Developer Kit, a new directory called `WEB-INF` is created within this directory. The `WEB-INF` directory contains some configuration information for Tomcat as well as a directory into which servlet class files are placed for deployment.

As long as the JSP file ends in the .jsp extension, you can simply load a JSP page in the `/home/groups/home/web` directory and the Web server knows to pass it to Tomcat for processing.

For example, if you have a file named `test.jsp`, you would use an FTP client or file sharing to load the file in the directory

```
/home/groups/home/web/
```

You would then access the dynamic JSP page through the URL

http://[hostname_of_server]/test.jsp

The first time you access this page, Tomcat takes a few seconds to compile the page but subsequent accesses are much quicker.

Compiling a servlet source Java file creates a servlet class file.

Deploying a servlet on the Sun Cobalt Qube 3 appliance is as simple as loading the servlet class file in the directory

```
/home/groups/home/web/WEB-INF/classes/
```

For example, if you have a file named `test.class`, you would use an FTP client or file sharing to load the file in the directory

```
/home/groups/home/web/WEB-INF/classes/test.class
```

The servlet is executed through the URL

http://[hostname]/servlet/test

Accessing this page is fast because the page does not need to be compiled.

Note: If you have a source file for a Java™ servlet, see “Compiling a Java™ servlet source file” on page 9.

You can also deploy JSPs and servlets through a Java .war file, which is a convenient way to deploy a Web application. For more information, “Deploying a .war file” on page 7.

Deploying JSPs and servlets on a Sun Cobalt RaQ server appliance

A Sun Cobalt RaQ server appliance can host multiple virtual sites. The Sun Cobalt Java Developer Kit creates a default Tomcat context for each virtual site in the Web directory for each individual site:

```
/home/sites/[sitename]/web/
```

A new directory called `WEB-INF` is created within this directory. The `WEB-INF` directory contains some configuration information for Tomcat as well as a directory into which servlet class files are placed for deployment.

As long as the JSP file ends in the .jsp extension, you can simply load a JSP page in the `/home/sites/[sitename]/web` directory and the Web server knows to pass it to Tomcat for processing.

For example, if you have a file named `test.jsp`, you would use an FTP client or file sharing to load the file in the directory

```
/home/sites/[site name]/web/
```

You would then access the dynamic JSP page through the URL

http://[sitename]/test.jsp

The first time you access this page, Tomcat takes a few seconds to compile the page but subsequent accesses are much quicker.

Compiling a servlet source Java file creates a servlet class file.

Deploying a servlet on a Sun Cobalt RaQ server appliance is as simple as loading the servlet class file in the directory

```
/home/sites/[sitename]/web/WEB-INF/classes/
```

For example, if you have a file named `test.class`, you would use an FTP client or file sharing to load the file in the directory

```
/home/sites/[sitename]/web/WEB-INF/classes/
```

The servlet is executed through the URL

http://[sitename]/servlet/test

Accessing this page is fast because the page does not need to be compiled.

Note: If you have a source file for a Java™ servlet, see “Compiling a Java™ servlet source file” on page 9.

You can also deploy JSPs and servlets through a Java™ .war file, which is a convenient way to deploy a Web application. For more information, “Deploying a .war file” on page 7.

You cannot execute JSPs and servlets on site user Web sites; execution is limited to a virtual site.

Web archive (.war) files

A Web archive (.war) file is a type of package file that integrates into a single file the JSP files, servlets and related support files that represent the Web tier (or possibly all tiers) of a Web application. A .war file allows for the easy deployment and portability of a Web application.

A .war file is created with the Java ‘jar’ command-line utility; this utility can be found in the J2SE or through a third-party development environment, such as Sun’s Forté™ set of tools. Custom scripts in the Sun Cobalt Java Developer Kit facilitate the deployment of new Web applications in their own context for Tomcat. This scheme is similar to the way a standard installation of Tomcat automatically deploys a Web application by simply loading .war files in the appropriate directory.

Creating a .war file

To create a .war file, start in a directory that contains a Web application based on JSP and servlet technology. The directory structure for it might look like the following example:

```
myapp/html/
myapp/images/
myapp/jsp/
myapp/WEB-INF/
myapp/WEB-INF/classes
myapp/WEB-INF/lib
```

Within the directory `myapp`:

- Your static content can be placed in the `html` and `images` directories.
- JSP pages can be stored in the `jsp` directory.
- `WEB-INF` is a special directory necessary for the execution of servlets; this directory is necessary if you want to use Java™ Beans or other Java™ classes in your application.
- Servlet class files and Java Beans are placed in the `WEB-INF/classes` directory.
- External Java classes can be placed in the `WEB-INF/lib` directory.

To make a .war file out of this structure on a Sun Cobalt server appliance, use the Java jar command-line utility:

1. Telnet into the server appliance as the user *admin*.
2. Change to the `myapp` directory.
3. Issue the following command:

   ```
   jar –cvf ../myapp.war
   ```

   This command creates a .war file and places it one directory above the `myapp` directory.

Deploying a .war file

To deploy a .war file, simply load the file into the root directory of the Web site of the Sun Cobalt server appliance to which you want to deploy it.

The Sun Cobalt Qube 3 appliance can host only one domain, so there is only one root Web site. It is located in the directory

```
/home/groups/home/web/
```

A Sun Cobalt RaQ server appliance can host several domains or virtual sites. The root web directory for a virtual site is located in:

```
/home/sites/[sitename]/web
```

Now restart the Web service (with a Sun Cobalt Qube 3 appliance, you have to restart Tomcat as well).

Restarting the Web service

Once the .war file is in the appropriate location, you must restart the Web service to deploy the application. In this case, Web service means both the Web server on the Sun Cobalt server appliance and the Tomcat engine.

On a Sun Cobalt Qube 3 appliance:

1. Telnet into the appliance as the user *admin*.
2. Execute the following command:

   ```
   su –c "/etc/rc.d/init.d/tomcat.init restart; /etc/rc.d/
   init.d/httpd reload"
   ```

   The system prompts you for the password of the user *admin*.
3. Enter the password.

On a Sun Cobalt RaQ server appliance:

1. Telnet into the server appliance as the user *admin*.
2. Execute the following command:

   ```
   su –c "/etc/rc.d/init.d/httpd reload"
   ```

   The system prompts you for the password of the user *admin*.
3. Enter the password.

When the Web service restarts, the system searches the Web roots for .war files. If one is found, a directory is created with the same name as the .war file (without the .war extension). The contents of the .war file are then extracted to that directory. Additionally, a new context for Tomcat is added so that Tomcat knows about your Web application.

For example, if a file named `myapp.war` is deployed on a Sun Cobalt RaQ server appliance in the directory `/home/sites/www.mydomain.com/web`:

- the directory `/home/sites/www.mydomain.com/web/myapp` is created
- the contents of `myapp.war` are extracted to this new directory
- a new context called myapp is added to Tomcat.

You can access the application is through the URL

http://www.mydomain.com/myapp/

Since a new context for Tomcat was added, servlets can be executed through the URL

http://www.mydomain.com/myapp/servlet/MyAppServlet

as long as the MyAppServlet servlet was deployed in the `WEB-INF/classes` directory of the .war file.

There are a two sample .war files distributed with the Sun Cobalt Java Developer Kit. These paths to the two files are:

```
/usr/java/jakarta-tomcat/webapps.inactive/examples.war
/usr/java/jakarta-tomcat/webapps.inactive/test.war
```

Compiling a Java™ servlet source file

An integrated development environment such as Sun's Forté set of tools may not be available for developing servlets. In that case, you can develop servlets "manually" through the command line of the UNIX® operating system and a text editor.

The Sun Cobalt Java Developer Kit installs a file called `java.sh` in `/etc/profile.d/`. When you log in to the server, your shell sources this file and some relevant Java-environment variables are set up for you. In particular, your `$CLASSPATH` (an environment variable in the UNIX operating system) is set up so that the Java™ compiler can locate the servlet classes.

To compile a Java™ servlet source file into a Java class file that Tomcat can use:

1. Telnet into the Sun Cobalt server appliance. Any user with shell access will work; you do not need to be the user *admin*.
2. Change to the directory in which the file is located.
3. Enter:

   ```
   javac myServlet.java
   ```

   Substitute your filename for `myServlet.java`.
4. If the compilation is successful, no errors are reported and a `myServlet.class` file is now found in the same directory.

To use this servlet:

1. Log in to the Sun Cobalt server appliance as a user with permissions to write in a Web root.
2. Load the class file in the `WEB-INFO/classes` directory of a Tomcat context.

For more information, see “Deploying JSPs and servlets on the Sun Cobalt Qube 3 appliance” on page 4 and “Deploying JSPs and servlets on a Sun Cobalt RaQ server appliance” on page 5.

Connecting to InterBase

Note: The InterBase database is available only on the Sun Cobalt Qube 3 appliance and the Sun Cobalt RaQ 4 and RaQ XTR server appliances. It is not available on the Sun Cobalt RaQ 3 server appliance.

The Sun Cobalt Qube 3 appliance and the Sun Cobalt RaQ 4 and RaQ XTR server appliances ship with the InterBase database.

InterBase 6.0 is an open-source, cross-platform SQL database from Inprise Corporation. InterBase is not enabled by default on the server appliances. InterBase offers a number of database features—triggers, stored procedures, blobs, event alerters, user-defined functions, multi-dimensional arrays, two-phase commit, referential integrity, constraints and a flexible set of transaction options.

For more information on InterBase, go to http://www.interbase.com.

The Sun Cobalt Java Developer Kit includes the InterClient driver that allows Java™ programs, JSPs and servlet applications to access to the InterBase database through the JDBC API. Tomcat has been pre-configured so that JSPs and servlets that call out for the driver are able to find it.

In order for programs to access the InterBase database, it must first be activated. The following paragraphs explain how to activate the InterBase database. Connecting to InterBase involves three steps:

1. Turn on the InterBase database server.
2. Assign a password to access the InterBase database server.
3. Create a database.

Turn on the InterBase server

If you have not already set up the InterBase database on your server appliance, perform the following steps to gain access to the database server.

1. Log into the Sun Cobalt server appliance as the user *admin*.
2. Enter

   ```
   su -
   ```

   The system prompts you for a password.
3. Enter the password for the user *admin*. You now have root privileges, meaning that you can change anything on the server appliances's operating system.
4. Use your preferred text editor (such as vi, emacs or pico) to edit the file `/etc/inetd.conf`.

   This is the configuration file for the `inetd` daemon.
5. Near or at the bottom of the file (probably the last line), there is a line that looks like:

   ```
   # interserver stream tcp nowait.100 root /usr/sbin/tcpd
   /usr/interclient/bin/interserver
   ```
6. Remove the hash mark (#) from in front of this line and save the file.

   The hash mark is a comment character and, when present, it disables the code that follows it. Removing the hash mark tells the inetd daemon to respond to network requests for the database.
7. Enter the following command:

   ```
   killall -HUP inetd
   ```

   This command tells `inetd` to reread its configuration file.

Assign a password to access the InterBase database server

Assign a password to access the database server. The simplest way is to do this is to use the existing *sysdba* account.

1. Enter the following command sequence:

   ```
   /opt/interbase/bin/gsec –database /opt/interbase/isc4.gdb
   ```

   A "GSEC >" prompt appears.

2. Enter:

   ```
   GSEC > modify sysdba –pw abc123
   ```

   This command changes the *sysdba* user password to *abc123*.

Note: Sun Microsystems, Inc. recommends using a password other than *abc123*.

3. Enter `quit` to leave the gsec utility.
4. Enter `exit` to leave the root account; you no longer need elevated privileges.

Create a database

Before you can begin creating tables and using the database in your application, you must first create a database. You can do this through the 'isql' command-line utility within InterBase which enables you to interact with the InterBase database server.

The following steps create a database file called `test.gdb` in the directory `/home/sites/home/`. You can put the database file in any location to which the user admin can write (since you are logged in as the user admin). If you are working on a Sun Cobalt Qube 3 appliance, you might want to put the database file in `/home/groups/home/`.

1. At the command line, enter:

   ```
   /opt/interbase/bin/isql –u sysdba –p abc123
   ```

   This command logs you in to the database server.

2. To create a database, enter:

   ```
   SQL> CREATE DATABASE "/home/sites/home/test.gdb";
   SQL> QUIT;
   ```

The InterBase database server is now ready to accept connections and can be manipulated using the *sysdba* account. For the database just created, you can access it using JDBC through the database URL:

```
jdbc:interbase://localhost/home/sites/home/test.gdb
```

The driver name for the InterClient driver is:

```
interbase.interclient.Driver
```

This name is also needed when using InterBase through JDBC.

To view some examples of Java code using the InterClient driver, go to:

```
/usr/interclient/examples
```

Additional documentation on InterClient can be found in the directory:

```
/usr/interclient/docs
```

Customizing Apache Tomcat

The standard, out-of-the-box configuration of Apache Tomcat may not fit your development needs. This is particularly true if your application requires elevated permissions, a custom JDBC driver or access to other Java classes stored on the server appliance.

Security Policy

All Tomcat contexts added by the Sun Cobalt Java Developer Kit receive a default set of security permissions. They are:

```
permission java.net.SocketPermission "localhost:1024-",
"listen,connect";
permission java.util.PropertyPermission "*", "read,write";
permission java.lang.RuntimePermission
"accessClassInPackage.sun.io";
permission java.io.FilePermission "-", "read,write,delete";
```

To add additional permissions:

1. Change to the directory

   `/usr/java/jakarta-tomcat/conf`

2. In this directory, create a file called

   `tomcat.policy.custom`

3. In this file, enter the security permissions that you want to add.

   The text in this file is appended to the default set of security permissions.

Note: Do not edit the `/usr/java/jakarta-tomcat/conf/tomcat.policy` file directly, as any changes are simply overwritten by the configuration scripts in the Sun Cobalt Java Developer Kit.

You must create the `tomcat.policy.custom` file which is then appended to the `tomcat.policy` file.

For example, if you want to allow all Tomcat contexts to have essentially "root" permissions, you would create the `tomcat.policy.custom` file and put the following text in it:

```
permission java.security.AllPermission;
```

Caution: Sun Microsystems, Inc. recommends against allowing all Tomcat contexts to have "root" permissions.

For more information on setting Tomcat permissions, see the Tomcat documentation at the URL http://jakarta.apache.org/tomcat.

Class path for database drivers and other classes

You may encounter a situation where the default class path for Tomcat does not suffice and you need to indicate the path for your own classes that exist elsewhere on the file system. Once Tomcat launches but before it begins channelling requests, it looks for the following file:

```
/etc/profile.d/tomcat.sh
```

If you want to add additional classes (for example, a new JDBC driver) to Tomcat's default CLASSPATH, add a statement like the following in the `tomcat.sh` file:

```
CLASSPATH=$CLASSPATH:/path/to/my/new/class
```

For the changes to take effect, you must restart the Web server and the Tomcat engine. See "Restarting the Web service" on page 8.

Disabling the Tomcat engine

If you no longer want to use Tomcat, you can deactivate it.

You must first turn off the Java virtual machines that handle the requests and prevents Tomcat from launching in the future.

To do this:

1. Log into the Sun Cobalt server appliance as the user *admin*.
2. Enter

   ```
   su -
   ```

   The system prompts you for a password.
3. Enter the password for the user *admin*. You now have root privileges, meaning that you can change anything on the server appliances's operating system.
4. Enter the following commands:

   ```
   /sbin/chkconfig –del tomcat.init
   /etc/rc.d/init.d/tomcat.init stop
   ```

You must now edit the Apache configuration file so that requests for JSPs and servlets are not handled.

1. In the Apache configuration file, locate the line with "# Setup Tomcat". A few configuration directives follow this line; these directives differ between the Sun Cobalt Qube 3 appliance and Sun Cobalt RaQ server appliances.
2. Put a hash mark (#) at the start of each of these lines until the first blank line. The hash marks disable these directives.

Finally, for the changes to take effect, you must restart the Web server. Enter the following command:

```
/etc/rc.d/init.d/httpd restart
```