

rh9-centos4 apache-php-mysql-tomcat-mod_jk.txt
how to install Apache + php + mysql along with tomcat and mod_jk

and java connectivity to mysql and oracle in quick and dirty way
(original how to from www.dantullis.com for redhat 9)

for connecting php to oracle, visit my blog here
<http://am3n.profusehost.net/index.php?id=45>

I am testing it with centos 4 and worked. I assume it also worked on rhel4

and centos/rhel/fc3
for Fedora Core 4 I think Tomcat5 and mod_jk already available on base

repository.
and on Fedora Core 5 you will have apache, tomcat and ruby working side by

side.
They already gives the software, all you need to do is install and

configure it.
But first before that I'll write on configuring apache, php, mysql, tomcat

and mod_jk :)

1. Install your centos 4 with webserver apache, php, mysql and othersoftware

needed

2. Install Java from <http://java.sun.com/j2se/1.5.0/download.jsp>
chooseJDK

5.0 RPM. Mine is JDK 5.0 update 5 you can download here.

2.1. Switch user (su) to ROOT
su -

2.2. Change to directory where you download JDK and install the RPM
rpm -ivh jdk-1_5_0_05-linux-i586.rpm

You will have the following directory
/usr/java/jdk1.5.0_05

2.3. Modify /etc/profile
vi /etc/profile

2.4. append these lines

```
rh9-centos4 apache-php-mysql -tomcat-modjk.txt
JAVA_HOME="/usr/java/jdk1.5.0_05";
JAVA_BIN="$JAVA_HOME/bin";
CLASSPATH=".: $JAVA_HOME/lib";
PATH="$PATH: $JAVA_HOME: $JAVA_BIN";
```

```
export JAVA_HOME CLASSPATH PATH
```

2.5. To make sure the new profile setting working, Logout from system then

Login again.

3. Install mod_jk1.2

3.1. Download mod_jk 1.2 connector from <http://jakarta.tomcat.org> or from

here

3.2. Copy the connectors to Apache modules directory

```
# cp jakarta-tomcat-connectors-jk-1.2.10-linux-sles9-i386-
```

```
prefork.so/etc/httpd/modules/mod_jk.so
```

4. Install and configure Tomcat5

4.1. Download Tomcat5 from <http://tomcat.apache.org> or from here

4.2. extract Tomcat5

```
# tar -xzf jakarta-tomcat-5.5.9.tar.gz -C /usr/local
```

```
# cd /usr/local/
```

```
# mv jakarta-tomcat-5.5.9 jakarta-tomcat
```

4.3. Edit and append these lines to Apache config file to allow connection

from apache to Tomcat

```
# vi /etc/httpd/conf/httpd.conf
```

```
#
```

```
# Mod_jk settings
```

```
#
```

```
JkWorkersFile "/etc/httpd/conf/workers.properties"
```

```
JkLogLevel error
```

```
# myNewApp is optional, in case you want to place a new app of your own in
```

here

```
JkMount /myapp default
```

```
JkMount /myapp/* default
```

```
#LoadModule jk_module modules/mod_jk.so
```

```
rh9-centos4 apache-php-mysql-tomcat-modjk.txt
LoadModule jk_module modules/mod_jk.so
```

```
# End of mod_jk settings
```

4.4. add the following file workers.properties on apache configuration

folder

```
# vi /etc/httpd/conf/workers.properties
```

```
# workers.properties.minimal -
```

```
#
```

```
# This file provides minimal jk configuration properties needed to  
# connect to Tomcat.
```

```
workers.tomcat_home=/usr/local/jakarta-tomcat
```

```
worker.java_home=$JAVA_HOME
```

```
ps=/  
#
```

```
# The workers that jk should create and work with  
#
```

```
#worker.list=ajp13w  
worker.list=default
```

```
#
```

```
# Defining a worker named ajp13w and of type ajp13  
# Note that the name and the type do not have to match.  
#
```

```
worker.default.type=ajp13
```

```
#
```

```
worker.default.type=ajp13
```

```
worker.default.host=localhost
```

```
worker.default.port=8009
```

```
worker.default.lbfactor=1
```

4.5. Start the Tomcat server, you'll need to run Tomcat first before you

can access your java apps

```
# /usr/local/jakarta-tomcat/bin/catalina.sh start
```

to stop it type, (which only need it if you want to deploy war file but

don't forget to start it again)

```
# /usr/local/jakarta-tomcat/bin/catalina.sh stop
```

check the open ports (example on my machine)

```
# netstat -pant
```

```
tcp 0 0::ffff:127.0.0.1:8005 :::* LISTEN 3262/java
```

```
tcp 0 0:::8009 :::* LISTEN 3262/java
```

```
tcp 0 0:::8080 :::* LISTEN 3262/java
```

Now open in your webbrowser <http://:8080>

rh9-centos4 apache-php-mysql-tomcat-modjk.txt
i.e. http://10.10.105.105:8080
If you see the default Tomcat Homepage then you are successfully installing

java and tomcat.

4.6. Now lets testing the mod_jk configuration. If you added the lines

JkMount on /etc/httpd/conf/httpd.conf You'll also need to add this line on

to your tomcat server.xml file on host section just before the closing tag

```
# vi /usr/local/jakarta-tomcat/conf/server.xml
<context reloadable="true" debug="1" docbase="myapp" path="/myapp"
/>
```

my complete host section is below:

```
<Host name="localhost" appBase="webapps"
unpackWARs="true" autoDeploy="true"
xmlValidation="false" xmlNamespaceAware="false">
<Context path="/myapp" docBase="alterJava" debug="0" reloadable="true"
/>
```

Restart the Tomcat (catalina.sh stop and start) and Apache (service httpd

restart) to working with new configuration.

You'll need to append those 2 lines on httpd.conf and 1 line on server.xml

everytime you want to deploy java application which accessible without

typing the port number. in this example we can call

http://10.10.105.105/myapp

rather than

http://10.10.105.105:8080/myapp

4.7. Now we create the sample java app

```
# cd /usr/local/jakarta-tomcat/webapps/
# mkdir -p myapp/WEB-INF/classes myapp/WEB-INF/lib
# vi myapp/index.jsp
```

tomcat-apache working together

4.8. Open http://10.10.105.105/myapp , it should display the line

rh9-centos4 apache-php-mysql -tomcat-modjk.txt
tomcat-apache working together

5. Configuring your TOMCAT

5.1. Disable Port 8080 (Port 8080 is only used in stand-alone mode and

development only which is not integrated with Apache WebServer)

```
# vi /usr/local/jakarta-tomcat/conf/server.xml
<!-- Define a non-SSL Coyote HTTP/1.1 Connector on port 8080 -->
<!-- (Uncommented for Productive Environment)
<Connector className="org.apache.coyote.tomcat4.CoyoteConnector"
port="8080" minProcessors="5" maxProcessors="75" enableLookups="true"

redirectPort="8443" acceptCount="10" debug="0"
connectionTimeout="20000"

useURIValidationHack="false" />
-->
```

5.2. The next step is to tell Tomcat to check the modification dates of the

class files of requested servlets, and reload ones that have changed since

they were loaded into the server's memory. This slightly degrades performance in deployment situations, so is turned off by default. However,

if you fail to turn it on for your development server, you'll have to

restart the server every time you recompile a servlet that has already been

loaded into the server's memory. Since this tutorial discusses the use of

Tomcat for development, this change is strongly recommended.

To turn on servlet reloading, edit install_dir/conf/context.xml and change

```
# vi /usr/local/jakarta-tomcat/conf/context.xml
```

```
<Context>
```

to

```
<Context reloadable="true">
```

rh9-centos4 apache-php-mysql-tomcat-modjk.txt

5.3. The invoker servlet lets you run servlets without first making changes

to your Web application's deployment descriptor (i.e., the WEB-INF/web.xml file). Instead, you just drop your servlet into WEB-INF/classes and use the URL `http://host/servlet/ServletName` (or

`http://host/webAppName/servlet/ServletName`) once you start using your own

Web applications.

The invoker servlet is extremely convenient when you are learning and even

when you are doing your initial development.

You almost certainly want to enable it when learning, but you should disable it again before deploying any real applications.

To enable the invoker servlet, uncomment the following servlet and servlet

-mapping elements in `install_dir/conf/web.xml`. Do not confuse this Apache

Tomcat-specific web.xml file with the standard one that goes in the WEB-INF

directory of each Web application.

uncomment below sections

```
# vi /usr/local/jakarta-tomcat/conf/web.xml
```

```
<servlet>
  <servlet-name>invoker</servlet-name>
  <servlet-class>
    org.apache.catalina.servlets.InvokerServlet
  </servlet-class>
  <init-param>
    <param-name>debug</param-name>
    <param-value>0</param-value>
  </init-param>
  <load-on-startup>2</load-on-startup>
</servlet>

<servlet-mapping>
  <servlet-name>invoker</servlet-name>
  <url-pattern>/servlet/*</url-pattern>
</servlet-mapping>
```

rh9-centos4 apache-php-mysql-tomcat-modjk.txt

5.4. Add connectivity to MySQL database. Download [mysql module here](#). and put

```
it on folder /usr/local/jakarta-tomcat/common/lib/  
# cp mysql-connector-java-3.0.9-stable-bin.jar/usr/local/jakarta-  
tomcat/common/lib/  
Restart tomcat to recognize new library.
```

Add connectivity to Oracle database. I'm using Oracle DB 10g. Download

```
oracle module here. and put it on folder /usr/local/jakarta-  
tomcat/common/lib/  
# cp ojdbc14.jar /usr/local/jakarta-tomcat/common/lib/
```

Update mysql module which you can download [here](#)
cp mysql-connector-java-3.1.12-bin.jar /usr/local/jakarta-
tomcat/common/lib/
Restart tomcat to recognize new library.

5.5. Add this line to /etc/rc.local to load Tomcat when your server is

```
turned on  
# vi /etc/rc.local  
/usr/local/jakarta-tomcat/bin/catalina.sh start
```

6. Done