Installing Multiple Instances of Tomcat 6

With Internet Information Services

Prepared by Piper McKinnon

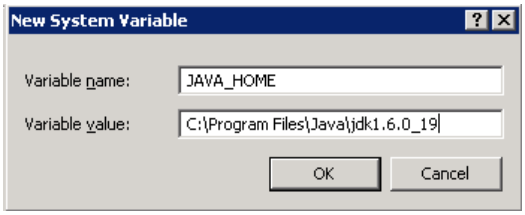
For the NPCA

November 16, 2010

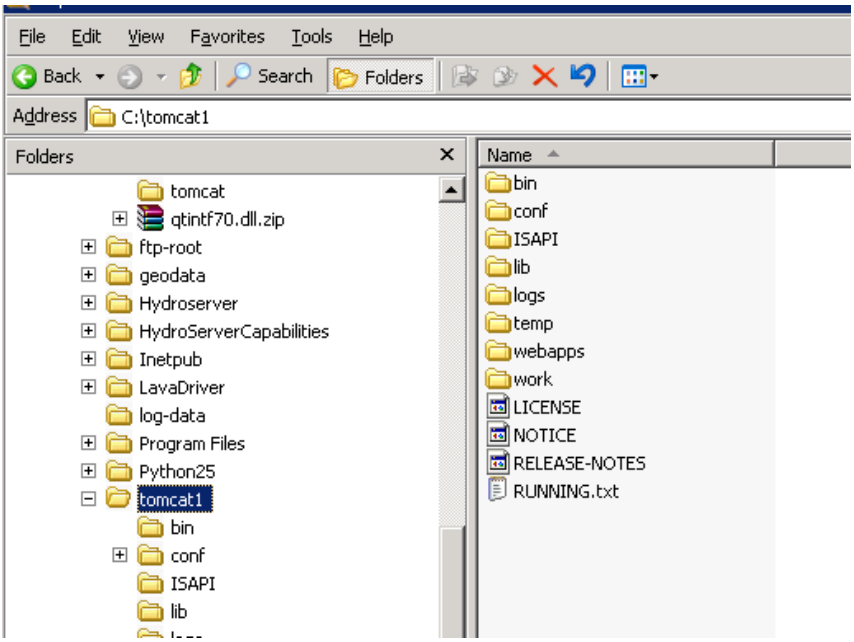
Contents

Installing the Initial Instance

1. Download and install latest Java JRE from http://www.java.com/en/download/manual.jsp if it isn't already installed
2. Set the Java_Home environment variable by going to the properties for 'My Computer' → Advanced and clicking the 'Environment Variables' button. Make sure JAVA_HOME isn't already in the system variables list. If not, click 'New' under system variables and add it like so, where the variable value is the full path to your jre or jdk installation:



3. Download the tomcat installation archive *DO NOT use the windows installer when the intent is to have multiple Tomcats. See http://tomcat.apache.org/whichversion.html to understand versions and installation packages, then go to the appropriate download page. For Windows Server 2003, use Windows zips (32/64 bit depending on your machine).
4. Unpack the zip file to your directory path of choice. For example C:\tomcat1. Once unpacked, your directory should look like this:



5. Tomcat is already technically installed now, but we need to run it as a service. In order to do that we can run the service.bat file located in the bin directory
    a. Go to Start → Run and type cmd to open the command prompt
    b. Change directory to C:\Tomcat1\bin

c. Type 'service.bat install [your service name]' (without the quotes) where 'your service name' is a name of your choice (no spaces). If you do not specify a service name, it will assign the default which is Tomcat. For this example we are calling it tomcat1.

6. Once the batch file has run successfully, you can exit the command prompt
7. Go to Start → Administrative Tools → Services and you should see your service listed as 'Apache Tomcat [your service name]'. Right click it and go to properties.
8. It's a good idea to run a service from a user account as opposed to the system login. To do this we need to create the user
    a. Go to Start → Administrative Tools → Computer Management →Local Users and Groups→Users
    b. Right-click and add a new user
    c. Give the user a username, description and password
    d. Make sure to uncheck 'User must change password at next logon'
    e. Now we need to give the user rights to both tomcat and java
    f. Go to the directory properties for your tomcat installation, and go to the Security tab. Add your new user and give them full control rights to the directory.
    g. Now go to the directory properties for your jre/jdk install and repeat the same steps, providing read and execute rights instead of full access
    h. Now we need to adjust the account settings to make it more secure. Go to Administrative Tools > Local Security Policy > Local Policies > User Right Assignments and add the user to the following rules:
        i. Deny Logon Locally
        ii. Deny Access to This Computer from the Network
        iii. Deny Logon As a Batch Job
        iv. Deny Logon Through Terminal Services
        v. Log on as a Service
    i. Now go back to Services and open the properties for your tomcat service and go to the Log On tab. Change the settings to use 'this account' and enter your user's login
9. Now go to the General tab and change the startup type to Automatic.
10. Restart the service. The first instance of Tomcat is now up and running. To test it, open a web browser and go to http://localhost:8080. If the Tomcat default home page opens, the installation was a success!
11. Now a few cleanup and optimization steps
    a. If you did not use the default name for your service name, you should browse to the tomcat1 bin directory and rename the file called tomcat6w.exe. to [yourinstancename]w.exe, ie. tomcat1w.exe (the 6 in the original file name represents the version number, so if you installed a different version it may be ie. tomcat7w.exe). This executable is for the tomcat manager application, and if it is not named with your service name it will fail on startup.
    b. Now to make sure you can access the Tomcat Manager, open the file C:\tomcat2\conf\tomcat-users.xml in a text editor. There should be two elements that looks like this:

<role rolename="manager" />
<role rolename="admin" />

If not, add them. Then check to see if there is user with the manager and admin roles. If there is, change the username and password attributes to your login values of choice. If not, add this line and assign the username and password of your choice:
<user username="tomcat" password="tomcat" roles="manager,admin" />

You must restart the service for the changes to effect (this is the case with all Tomcat configuration files). Now you can go back to http://localhost:8080 and click on the Status or Tomcat Manager links in the Administration menu. You will be prompted to enter your login. Within these areas you can check the performance of the tomcat server and any applications you are running on it.

c. Now double click tomcat1w.exe and go to the Java tab. Java has a number of memory settings that by default are set impractically low for production environments. This can cause OOM (out of memory) errors when running applications. Check the initial memory pool and maximum memory pool settings. The maximum setting should never be more than half of the RAM on the computer, and a good rule of thumb is to set the initial memory to half of your maximum. Click apply and ok after making any changes. Restart the tomcat service for these changes to take effect.
d. Further Java optimization can be done in the options section of the Tomcat manager Java tab, or by placing them in a batch file called setenv.bat. This is practical when running multiple instances as the file can simply be copied from the bin directory of one instance to another. Run the file from the command prompt. See Appendix VI for an example setenv.bat configuration.

Installing Additional Instances

1. The process to install a second instance is essentially the same, however we need to make some configuration changes to ensure that the new instance is running on a different port number.
2. You can use the same zip archive you downloaded for the first instance. Unpack the contents to a new directory, for example C:\Tomcat2
3. Now, start it as a service by going to Start → Run and typing cmd. Change the directory to C:\Tomcat2\bin and then type 'service.bat install [your service name]'. This time we must specify the service name as using the default would cause it to overwrite our initial instance (if we used the default name for that instance).
4. Go to Start→Administrative Tools→ Services and go to the properties of your new service. This time we do not need to create a new user to run the service, we can just assign the user created for when configuring the first instance. Go to the Log On tab, choose the 'this account' option and enter the username and password.
5. Now, because we already have one tomcat running on the default port, we need to make some configuration changes to this new instance, otherwise neither would work properly. Open the file C:\tomcat2\conf\server.xml in a text editor.
6. We need to modify the shutdown, HTTP and AJP connectors ports. Find the following lines:

   ```
   <Server port="8005" shutdown="SHUTDOWN">
   <Connector port="8080" protocol="HTTP/1.1" connectionTimeout="20000"
   redirectPort="8443" />
   <Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />
   ```

   And alter the port and redirect port numbers to read something like:

   ```
   <Server port="8105" shutdown="SHUTDOWN">
   <Connector port="8180" protocol="HTTP/1.1" connectionTimeout="20000"
   redirectPort="8543" />
   <Connector port="8109" protocol="AJP/1.3" redirectPort="8543" />
   ```

   If you are not sure what ports are already in use, you can go to the command prompt and type netstat –a. This will list all ports already listening.

7. Go back to the Services interface, change the startup type to Automatic and restart the service.
8. You should now be able to browse to http://localhost:8180 and see the second tomcat default page.
9. Perform the same optimization and cleanup actions as in step 11 of the initial installation.

Installing the ISAPI Redirector with multiple Tomcats for IIS

The ISAPI redirector is required if you want to run Tomcat on an IIS machine and have IIS handle and redirect incoming requests to your IP to the appropriate applications. Installing the redirector is a cinch, but configuring the properties files can be a challenge depending on your setup.

You can create as many workers as you like in the workers.properties files. For multiple instances of Tomcat, you should use a separate load balancing worker for each instance. In theory this should not be necessary, but in our case, it was impossible to get the second Tomcat to run without its own balancer.

Every web application must be mapped to a worker (in our case, load balancing workers) in the uriworkermap.properties file. Make sure you map your applications to the correct worker for the instance your app is running on.

1. Download Tomcat isapi connector from http://tomcat.apache.org/download-connectors.cgi (pre-built .dll is easiest to deal with)
2. Create a folder under C:\Tomcat1 (or whatever your initial instance is named) called ISAPI and unpack the tomcat connector isapi_redirect[release].dll to it. Rename the dll isapi_redirect.dll
3. Create a file called isapi_redirect.properties in a text editor and add the content in the example in Appendix I of this document, altering the paths to match your installation. Save this file to the ISAPI directory.
4. Go to the Tomcat1\conf directory and check to see if two files exist: workers.properties and uriworkermap.properties. If they don't, create them using the content in:
    a. IF YOU HAVE ONE TOMCAT INSTANCE: Appendices II and IV
    b. IF YOU HAVE MULTIPLE TOMCAT INSTANCES: Appendices III and V
5. To allow for Tomcat to pass requests to our web applications, we need to add mappings to them in the uriworkermap.properties file. Every web application must be mapped to a worker (in our case, load balancing workers). Make sure you map your applications to the correct worker for the instance your app is running on. If using the example configurations, any application on tomcat1 would be mapped like this: /[yourappname]/*=wlb while applications on tomcat2 would look like this: /[yourappname]/*=nlb
   *NOTE: ensure that the worker port number matches the http connector port for your Tomcat instance using that worker.
   **NOTE: Tomcat configuration files are read on startup, so any edits to them require the service to be restarted.

Configuring IIS to Use the ISAPI Redirector

1. IIS has to be configured to pass java requests to Tomcat
2. Open IIS manager and create a new virtual directory under the website that will host applications (default or otherwise). Name the virtual directory Jakarta
3. For the content directory, enter the path to the ISAPI directory
4. For permissions, allow read, run and execute. Complete the wizard
5. Now, go to the properties of the website and choose the ISAPI Filters tab
6. Add a filter, name it Tomcat ISAPI, and point it to the isapi_redirect.dll
7. Now add a new web service extension in IIS to enable the ISAPI dll. Name the extension Tomcat ISAPI, and point it to isapi_redirect.dll. Make sure 'set extension to Allowed' is checked

Appendix I – Example Isapi_redirect.properties

```
#config file for the Jakarta ISAPI redirector
#The path to the ISAPI redirector Extension, relative to the website
#This must be in a virtual directory with execute priveleges
extension_uri=/jakarta/isapi_redirect.dll

#Full path to the log file for the ISAPI redirector
log_file=C:\Tomcat1\logs\isapi_redirect.log

#Log level (debug, info, warn, error or trace)
log_level=info


#full path to the workers.properties file
worker_file=C:\Tomcat1\conf\workers.properties

#full path to the uriworkermap.properties file
worker_mount_file=C:\Tomcat1\conf\uriworkermap.properties
```

Appendix II – Example workers.properties for One Tomcat

```
# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements.  See the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
# http://www.apache.org/licenses/LICENSE-2.0
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# workers.properties.minimal -
# This file provides minimal jk configuration properties needed to connect to Tomcat.
#
# The workers that jk should create and work with
#
worker.list=wlb,jkstatus

# Defining a worker named node1 and of type ajp13
# Note that the name and the type do not have to match.
#
worker.node1.type=ajp13
worker.node1.host=localhost
worker.node1.port=8009
worker.node1.lbfactor = 1

# Defining load balancers
worker.wlb.type=lb
worker.wlb.balance_workers=node1

# Define status worker
#
worker.jkstatus.type=status
```

Appendix III – Example workers.properties for Two Tomcats

```
# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements.  See the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
# http://www.apache.org/licenses/LICENSE-2.0
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# workers.properties.minimal -
# This file provides minimal jk configuration properties needed to connect to Tomcat.
#
# The workers that jk should create and work with
#
worker.list=wlb,nlb,jkstatus

# Defining a worker named node1 and of type ajp13
# Note that the name and the type do not have to match.
#
worker.node1.type=ajp13
worker.node1.host=localhost
worker.node1.port=8009
worker.node1.lbfactor = 1

# Defining a worker named node2 and of type ajp13
# to handle requests on Tomcat2
#
worker.node2.type=ajp13
worker.node2.host=localhost
worker.node2.port=8109
worker.node2.lbfactor = 1

# Defining load balancers
worker.wlb.type=lb
worker.wlb.balance_workers=node1
worker.nlb.type=lb
worker.nlb.balance_workers=node2

# Define status worker
#
worker.jkstatus.type=status
```

Appendix IV – Example uriworkermap.properties for One Tomcat

```
# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements.  See the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
# http://www.apache.org/licenses/LICENSE-2.0
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# uriworkermap.properties - IIS
#
# This file provides sample mappings for example wlb
# worker defined in workermap.properties.minimal
# The general syntax for this file is:
# [URL]=[Worker name]

#admin and manager only needed if you want to access other than on the localhost
/admin/*=wlb
/manager/*=wlb
/examples/*=wlb
/webAppOnTom/*=wlb

# Optionally filter out all .jpeg files inside that context
# For no mapping the url has to start with exclamation (!)
!/servlets-examples/*.jpeg=wlb

# Mount jkstatus to /jkmanager
# For production servers you will need to
# secure the access to the /jkmanager url

/jkmanager=jkstatus
```

Appendix V – Example uriworkermap.properties for Two Tomcats

```
# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements.  See the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
# http://www.apache.org/licenses/LICENSE-2.0
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# uriworkermap.properties - IIS
#
# This file provides sample mappings for example wlb
# worker defined in workermap.properties.minimal
# The general syntax for this file is:
# [URL]=[Worker name]

#admin and manager only needed if you want to access other than on the localhost
/admin/*=wlb
/manager/*=wlb
/examples/*=wlb
/webAppOnTom1/*=wlb
/webAppOnTom2/*=nlb

# Optionally filter out all .jpeg files inside that context
# For no mapping the url has to start with exclamation (!)
!/servlets-examples/*.jpeg=wlb

# Mount jkstatus to /jkmanager
# For production servers you will need to
# secure the access to the /jkmanager url

/jkmanager=jkstatus
```

Appendix VI – Example setenv.bat

set JAVA_OPTS=-server -Xmx1024m -Xms512m -XX:SoftRefLRUPolicyMSPerMB=36000 -XX:MaxPermSize=512m -XX:+CMSClassUnloadingEnabled --XX:+CMSPermGenSweepingEnabled

Note these setting can also just be added in the java options section on the java tab of the tomcat config tool (found in the bin folder ie. tomcat1w.exe)

Troubleshooting

Dealing with the Out of Memory (OOM) Errors

This is a very common problem with Tomcat. Out of Memory errors mean just that, you don't have enough memory to service the demand on all of your web apps. If you have more memory available to allocate than the default assigned, it may be just a matter of increasing the heap size (see section 11 c of 'Installing the Initial Instance'). If that doesn't fix the problem, there may be memory leaks occurring with one or more of your applications. These can be frustrating to troubleshoot, as there likely isn't anything really 'wrong' with your code; it just isn't optimized well enough to play nice with the JVM.

OutOfMemory errors are logged in the 'stdout_[datestamp].log' file. The easiest way to find the cause of the errors is to use a web app profiler like YourKit, Eclipse Memory Analyser or a comparable open source package (a good list can be found here: http://java-source.net/open-source/profilers).

There are numerous other Java settings that can be enabled to help alleviate the issue (as demonstrated in Appendix VI) if code optimization does not solve the problem, or if altering code is simply not an option (ie. third party apps, lack of technical ability). This website has an exhaustive list of available Java options: http://www.oracle.com/technetwork/java/javase/tech/vmoptions-jsp-140102.html.