5.0 Zero Footprint Tomcat Setting Up Multiple Instances

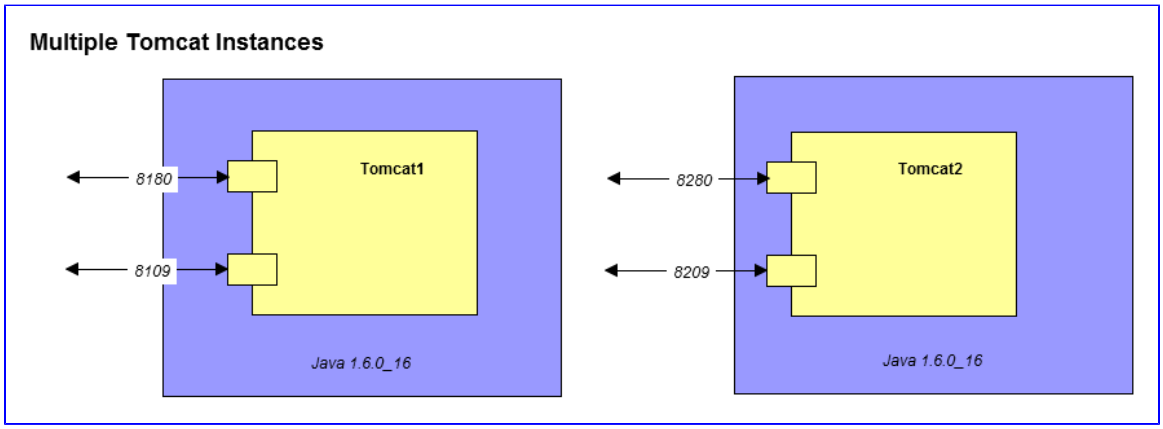
Introduction

This is a continution of the Bonsai Framework 0FS Tomcat Setup and explains how to scale vertically adding two additional instances of Tomcat on the same machine,

- 0FS Tomcat Instance 1 on Java 1.6.0_16 run by user serveradmin
- 0FS Tomcat Instance 2 on Java 1.6.0_16 run by user serveradmin

You may also opt to further mix and match with different versions of Tomcat and Java.

Here is a diagram of what we will be building,



Download

Download the Bonsai Framework pre-packaged 0FS Tomcat setup.

Install by unzipping with a staff sudo enabled account,

```
wget
http://www.bonsaiframework.com/downloads/0fs-tomcat-linux/0fs-tomcat.tar.gz
sudo tar -xvpf 0fs-tomcat.tar.gz # sudo will ensure the permissions and
users are kept
sudo mv /0fs-tomcat/ /0fs-tomcat.1/
sudo mv /0fs-tomcat.1/ /opt/
```

Create the second instance,

```
sudo tar -xvpf 0fs-tomcat.tar.gz # sudo will ensure the permissions and
users are kept
sudo mv /0fs-tomcat/ /0fs-tomcat.2/
sudo mv /0fs-tomcat.2/ /opt/
```

These two Tomcat instances are completely independent of each other.

Change Default Ports

The different Tomcat instances cannot be listening on the same port.

Default ports for Tomcat are,

- 8005 - for shutdown
- 8009 - for JK Mod
- 8080 - regular port similar to 80
- 8443 - ssl port similar to 443

Tomcat1

The new ports for **Tomcat1** will be,

- 8105 - for shutdown
- 8109 - for JK Mod
- 8180 - regular port similar to 80
- 8543 - ssl port similar to 443

Verify that the ports for Tomcat1 are **not** being used.

```
netstat -an | grep LISTEN | grep 8105
netstat -an | grep LISTEN | grep 8109
netstat -an | grep LISTEN | grep 8180
netstat -an | grep LISTEN | grep 8543
```

You should not see anything return from running the above commands.

Change Tomcat1 to use use the new ports by editing **/opt/0fs-tomcat/conf/server.xml**. Use an editor or more quickly using the sed commands,

```
cd /opt/0fs-tomcat.1/conf/
sed -i 's/8005/8105/' server.xml
sed -i 's/8009/8109/' server.xml
sed -i 's/8080/8180/' server.xml
sed -i 's/8443/8543/' server.xml
```

Tomcat2

Verify that the ports for Tomcat2 are **not** being used.

```
netstat -an | grep LISTEN | grep 8205
netstat -an | grep LISTEN | grep 8209
netstat -an | grep LISTEN | grep 8280
netstat -an | grep LISTEN | grep 8643
```

You should not see anything return from running the above commands.

Change Tomcat2 to use use the new ports by editing **/opt/0fs-tomcat/conf/server.xml**. Use an editor or more quickly using the sed commands,

```
cd /opt/0fs-tomcat.2/conf
sed -i 's/8005/8205/' server.xml
sed -i 's/8009/8209/' server.xml
sed -i 's/8080/8280/' server.xml
sed -i 's/8443/8643/' server.xml
```

Verify

Startup your Tomcat instances and verify that they are listening,

```
su - serveradmin

cd /opt/0fs-tomcat.1/bin/
./startup.sh
cd /opt/0fs-tomcat.2/bin/
./startup.sh
```

If you have a firewall setup **do not forget** to open the ports for testing and then close them afterwards if you plan to front with the Apache Web Server.

As covered in 0FS Tomcat Setup load a web browser to test, but replace 8080 with **8180 for Tomcat1** and **8280 for Tomcat2**.

Naming Convention of Tomcat Process for Solaris (solved)

Solaris' built in ps has a specific limitation of 60 characters. As a result, it is not intuitive to determine which process of tomcat is which when the ps command is executed,

```
serveradmin 12150 13290   1 11:51:28 pts/2       0:10
/opt/jre1.6.0_16/bin/java
-Djava.util.logging.manager=org.apache.juli.ClassLoad
serveradmin  5906 13290   0   Jul 24 ?           6:17
/opt/jre1.6.0_16/bin/java
-Djava.util.logging.manager=org.apache.juli.ClassLoad
```

Notice that the names are exactly the same.

Using the Bonsai Framework method of bundling Java with Tomcat we **can** tell the differences,

```
serveradmin 12150 13290   1 11:51:28 pts/2       0:10
/opt/0fs-tomcat.1/java
-Djava.util.logging.manager=org.apache.juli.ClassLoad
serveradmin  5906 13290   0   Jul 24 ?           6:17
/opt/0fs-tomcat.2/java
-Djava.util.logging.manager=org.apache.juli.ClassLoad
```

Next to Cloud

I've still to write this, but I hope how folks can see how this will fits nicely into building Cloud friendly containers using LXC or Docker.