

# TomC@ - HowTo's

by Peter Roßbach

## 1. Tomcat Server Template: webdevPLUS

Peter Roßbach

### 1.1. Anspruch an eine flexible Tomcat Konfiguration

Die Tomcat-Konfiguration ist mittlerweile so flexibel geworden, dass nur noch alte Hasen in der Lage sind, alle Einstellungsmöglichkeiten auf Anhieb zu überblicken. Dies führt in der Praxis leider auch im produktiven Umfeld häufig - wieder besseren Wissens - zur Verwendung der Tomcat Standardinstallation inklusive der Standard Projektkonfiguration. Dabei steht bereits seit den frühen Tagen des Tomcat 4 Servers mit der Trennung von Projektkonfiguration auf der einen und Server Release auf der anderen Seite ein mächtiges Werkzeug zur Verfügung, welches das Hinterlegen mehrerer getrennter Konfigurationen für unterschiedliche Projekte erlaubt. Grund genug für uns, dieses Konzept einmal näher mitsamt einer eigenen, praktischen Erweiterung vorzustellen.

Die vorliegende Beschreibung könnte unter dem Motto *Eine praktische Anleitung zur flexiblen Tomcat-Konfiguration* stehen. Wer kennt nicht den Wunsch, mal eben schnell eine Tomcat Konfiguration zu erstellen, um so die eine oder andere neue Einstellung und deren Auswirkungen für ein Projekt testen zu können. Natürlich sollen diese Änderungen nicht zu Lasten anderer Projekte gehen. In der Praxis sieht die Lösung dann häufig so aus, dass ein neuer Tomcat Server parallel zu dem bzw. den bisherigen Tomcat Servern installiert wird und die gewünschten Einstellungen an der im Standardumfang des Tomcat enthaltenen Konfigurationsdatei `$CATALINA_HOME/conf/server.xml` vorgenommen werden. Bei diesem Vorgehen wird in der Regel die Tatsache vernachlässigt, dass es sich bei der Standard `server.xml` lediglich um ein Beispiel für das Server Release handelt, nicht aber um eine optimierte Ausgangsbasis für die Konfiguration eines produktiven Systems. Auf vielen derart konfigurierten Servern kann man daher neben den eigentlich Angedachten Anwendungen zusätzlich die Tomcat Dokumentation oder gar die Tomcat Beispiele erreichen. Ein solches Szenario ist eigentlich immer ein Zeichen dafür, dass es an der Zeit ist in die Hände zu spucken und über eine bessere Alternative nachzudenken.

Genau eine solche Alternative möchten wir Ihnen im Verlauf dieser Kolumne vorstellen.

Dabei werden wir ausgehend von einer normalen Tomcat Installation zeigen, wie Sie Schritt für Schritt eine Unabhängigkeit von Server Release und Projektkonfiguration erreichen. Aus dieser Eigenschaft heraus erwachsen starke Konzepte für die Unabhängigkeit und Lebendigkeit ihrer Tomcat Server. Wäre es nicht wunderbar mit einem Skript einfach die nächste Konfiguration für ein Experiment oder das nächste Projekt generieren zu können? Diese Kolumne beschreibt einen Weg und natürlich ein Beispiel ihre Tomcat Konfigurationen zu säen und anschließend unabhängig voneinander wachsen zu lassen.

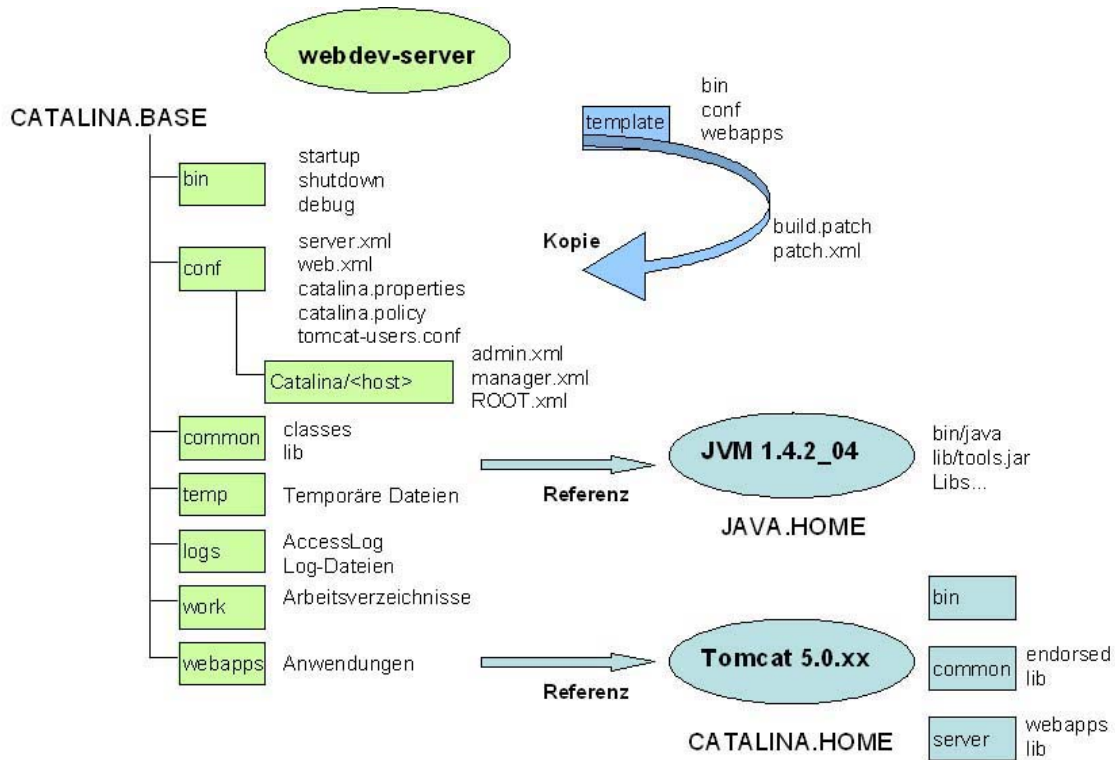
**Wer im Frühjahr nicht reichhaltig sät, kontinuierlich düngt und pflegt, wird keine reichhaltige Ernte einfahren.**

So oder ähnlich beginnen viele Bauernsprüche und dies erschien auch mir, im Laufe der diversen Tomcat-Projekte, als eine wesentliche Erkenntnis zu reifen. Leider wie es in Ihrem und unserem Alltag der Projekte schnell passiert, habe ich das berühmte berüchtigte **Copy-Paste** Antipattern, wieder besseren Wissens, jedes Mal wieder erfolgreich angewendet. Mittlerweile gibt es genug Beispiele, um über eine bessere Strategie nachzudenken. Ich habe Ihnen daher ein Saat-Muster für Tomcat Konfiguration geschaffen.

## 1.2. Die Konfiguration des webdevPLUS

Um möglichst unabhängig von einem Tomcat-Release zu bleiben, bietet es sich an das Release unberührt zu lassen und nur auf ein solches zu verweisen. Der Tomcat 4 führt diese Konfigurationsoption das **CATALINA\_BASE** ein. Die Systemproperties *-Dcatalina.home* verweist auf die Binaries einer Tomcat Releases und mit verschiedene *-Dcatalina.base* Verzeichnissen können getrennte Konfiguration auf der Basis eines Release hergestellt werden. Hierfür ist eine besondere Verzeichnisstruktur erforderlich, wie sie in der unteren Tabelle zu finden ist. Der grosse Unterschied zu einer klassischen Konfiguration eines Tomcat-Releases besteht nun darin, dass in den eigenen Startup-Skripten das eigene Verzeichnis als *catalina.base* angegeben wird und nur das *catalina.home* auf das Tomcat-Release verweist. Im Tomcat 5 ist es nun sogar möglich den Klassenpfad innerhalb einer *catalina.base* Konfiguration zu erweitern.

Als Besonderheit meines webdevPLUS ist die Anpassbarkeit an verschiedene Umgebungen durch Templates und die Saatfähigkeit zu nennen. Mein webdevPLUS können Sie also zentrale auf jeder Maschine installieren, verwalten und in enem beliebigen Verzeichnis für Ihre Experimente oder Projekte expandieren. Im zentralen *seed*-Verzeichnis auf jedem Rechner können die Patch Werte für Ihre Systemweites webdev-server Template hinterlegt werden. Natürlich können Sie in der jeweiligen Installation des webdev-server PLUS später die Templates oder Anpassung individuell ändern. Nach einer zentralen Änderung könne Sie das Template jederzeit wieder auf ein Beispiel anwenden.



## Download:

Mein **webdevPLUS** Template für den Tomcat 5, ist natürlich direkt zu haben. Nur noch ein Klick sind Sie von einer sinnvollerer Konfiguration auf Ihrem Tomcat Computer entfernt. Das Template ist unter Windows 2000, Windows XP, Mac OS X und Suse 8.2/9.0 getestet.

## webdevPLUS Template in der Version 1.1.6

(<http://tomcat.objektpark.org/examples/webdevplus.zip>)

## 2. Installation

- Voraussetzung ist die Installation eines Tomcat 5.0.19 oder höher, Ant 1.5.4 und ein JDK 1.4.x
- Download des **webdevPLUS**
- Auspacken des Zip-Archives in einem beliebigen Verzeichnis
- Öffnen einer Shell
- `cd TomC@_webdevplus_XX/`
- Editieren und Ändern der Datei `build.patch` oder entsprechende Plattformanpassung

```
java.home=D:/java/j2sdk1.4.2_03
```

```

catalina.home=D:/server/jakarta/jakarta-tomcat-5.0.19
seed.home=d:/seeds
server.shutdown.port=7305
server.shutdown.key=7305
server.debug=1
connector.http.port=7380
connector.ajp.port=7309
connector.https.port=7443
engine.host=localhost
jpda.address=8000
jvm.minmemory=8
jvm.maxmemory=32

```

- Die Beispieltemplates für die Systeme *Windows XP/2000/NT*, *LINUX*, *MAC OS X* befinden sich in den Dateien *build.[WIN/LINUX/MAC].patch*
- Die Anpassung an Ihre System nach der Änderungen der Datei *build.patch* geschieht mit *ant patch*
- Für verschiedene Systeme kann die Anpassung auch z.B. mit *ant -Ddeploytarget=LINUX patch* geschehen.
- Mit *ant start* kann Ihr Tomcat nun gestartet werden

Nun können Sie sich einfach mal die Template und Konfigurationen im Detail an sehen. Mit diesem typischen Geschmack auf mehr, können Sie nun meine Installationsweise nach dem Studium des nächsten Abschnitts, zentral zur Verfügung stellen und damit in Sekunden eine frische Tomcat-Konfiguration ihrer Wahl produzieren.

### 3. Verzeichnisse

#### 3.1. CATALINA.HOME: Verzeichnis der Tomcat Binary Distribution

```

jakarta-tomcat-5.0.xx
####bin
#   ####catalina.[sh|bat]
#   ####startup.[sh|bat]
#   ####shutdown.[sh|bat]
#   ####setclasspath.[sh|bat]
#   ####tomcat.exe
#   ####jsvc.tar.gz
#   ####bootstrap.jar
#   ####commons-logging-api.jar
#   ####commons-daemon.jar
#   ####jmx.jar (ab 5.0.20)
####common
#   ####endorsed
#   ####lib
####conf
#   ####Catalina
#   #       ####localhost
#   #       ####admin.xml

```

## TomC@ - HowTo's

```
# # #####manager.xml
# #####catalina.properties
# #####catalina.policy
# #####server.xml
# #####web.xml
# #####tomcat-users.xml
#####server
# #####lib
# #####webapps
# #####admin
# #####manager
#####webapps
#####balancer
#####jsp-examples
#####ROOT
#####servlets-examples
#####tomcat-docs
# #####appdev
# # #####sample
# # #####docs
# # #####src
# # #####web
# #####architecture
# # #####requestProcess
# # #####startup
# #####catalina
# # #####docs
# # # #####api
# # #####funcspects
# #####config
# #####jasper
# #####jspapi
# #####servletapi
#####webdav
```

Verzeichnis	Bedeutung
bin	Verzeichnis für alle Start und Stop-Skripte. Hier befindet sich das zentrale Steuerungsskript <i>catalina.sh</i> . Ebenso gibt es für die Integration der in die Systemdienste die entsprechende Unterstützung.
conf	Verzeichnis für die Tomcat-Konfigurationen, Default-Webkonfiguration Policy-Files, Keystores, etc.
conf/Catalina/localhost	Hier befinden sich die <b>context.xml</b> aller Anwendung. In der Standarddistribution sind dies <i>admin.xml</i> und <i>manager.xml</i> die auf die konkreten Anwendung im <i>\$catalina.home/server/webapps</i> Verzeichnis verweisen.

logs	Verzeichnis für Log-Dateien des Tomcat. Hier befinden sich das Access Log und die Protokollausgaben des Servers.
temp	Temporäres Tomcat-Verzeichnis
webapps	Default-Anwendungs-Verzeichnis mit der Dokumentation und den Beispielen
work	Arbeitsverzeichnis für kompilierte Dateien, etc.
common/endorsed und common/lib	Hier können in den Unterverzeichnissen <i>classes</i> Klassen und in <i>lib</i> jar Archive für die Erweiterung des common-Classloader plziert werden (s. conf/catalina.properties).
server/classes und server/lib	Hier können in den Unterverzeichnissen <i>classes</i> Klassen und in <i>lib</i> jar Archive für die Erweiterung des server-Classloader plziert werden (s. conf/catalina.properties).

### 3.2. CATALINA.BASE: Verzeichnis des webdevPLUS

```

webdev-server
#build.properties
#build.patch
#buildLINUX.patch
#build.xml
#patch.xml
####bin
####common (Optional)
#   ####classes
#   ####lib
####conf
#   ####Catalina
#   #   ####localhost
#   #   #   #...xml
#   #   #   #manager.xml
#   #   #   #admin.xml
#   ####server.xml
#   ####web.xml
#   ####tomcat-users.xml
####logs
####shared (Optional)
#   ####classes
#   ####lib
####temp
####template
#   ####bin
#   ####conf

```

## TomC@ - HowTo's

```
# #####webapps
####webapps
# #####ROOT
#     #####images
#     #####tiles
#     #####WEB-INF
#     #####lib
#     #####tlds
####work
```

Verzeichnis	Bedeutung
/	In der Datei <i>build.patch</i> können die Anpassung an Ihre Umgebung gemacht werden. Der Patch wird durch das Skript <i>patch.xml</i> aus dem <i>template</i> Verzeichnis heraus expandiert. Die Datei <i>build.xml</i> enthält alles nötige für die Laufzeitsteuerung des <i>webdev-server</i> Tomcats.
bin	Verzeichnis für alle Start und Stop-Skripte
conf	Verzeichnis für die Tomcat-Konfigurationen, Default-Webkonfiguration Policy-Files, Keystores, etc.
conf/Catalina/localhost	Hier befinden sich die <b>context.xml</b> aller Anwendung. In meinem Template sind dies mindestens <i>admin.xml</i> und <i>manager.xml</i> die auf die konkreten Anwendung im <i>\$catalina.home</i> Verzeichnis verweisen.
logs	Verzeichnis für Log-Dateien des Tomcat. Hier befinden sich das Access Log und die Protokollausgaben des Servers.
temp	Temporäres Tomcat-Verzeichnis
template	Hier befinden sich die Templates für die jeweiligen Anpassungen. Ich habe mich entschlossen auf die Skripte, <i>server.xml</i> und die Manager Anwendungen anzupassen.
webapps	Default-Anwendungs-Verzeichnis
work	Arbeitsverzeichnis für kompilierte Dateien, etc.
<i>Optional:</i> common/classes und common/lib	Hier können in den Unterverzeichnissen <i>classes</i> Klassen und in <i>lib</i> jar Archive für die Erweiterung des common-Classloader plaziert werden (s. <i>conf/catalina.properties</i> ).

*Optional:* shared/classes und shared/lib

Hier können in den Unterverzeichnissen *classes* Klassen und in *lib* jar Archive für die Erweiterung des shared-Classloader plziert werden (s. conf/catalina.properties).

## 4. Skripte

### 4.1. startup.bat

```
set JAVA_HOME=PATH-TO-JAVA-HOME
set CATALINA_BASE=..
set _CATALINA_HOME=%CATALINA_HOME%
set CATALINA_HOME=PATH-TO-CATALINA-HOME
set CATALINA_OPTS=-server -ms32m -mx64m
%CATALINA_HOME%\bin\catalina start %1 %2 %3 %4 %5 %6 %7 %8 %9
```

### 4.2. startup.sh

```
#!/bin/sh
#-----
#Start Script for the CATALINA ServiceServer
export CATALINA_BASE=..
export CATALINA_HOME=PATH-TO-CATALINA-HOME
export CATALINA_OPTS=-Xmx32m -Djava.awt.headless
exec $CATALINA_HOME/bin/catalina.sh start "$@"
```

### 4.3. debug.bat

```
@echo off
set _JAVA_HOME=%JAVA_HOME%
set _JSSE_HOME=%JSSE_HOME%
set JAVA_HOME=PATH-TO-JAVA-HOME
set CATALINA_BASE=..
set _CATALINA_HOME=%CATALINA_HOME%
set CATALINA_HOME==PATH-TO-CATALINA-HOME
set CATALINA_OPTS=-ms32m -mx64m -Dcatalina.config=file:../conf/catalina.properties
set JPDA_TRANSPORT=dt_socket
set JPDA_ADDRESS=8000
set CATALINA_OPTS=%CATALINA_OPTS% -Xdebug -Xrunjdpw:transport=%JPDA_TRANSPORT%, \
    address=%JPDA_ADDRESS%,server=y,suspend=n
echo CATALINA_OPTS: %CATALINA_OPTS% %CATALINA_HOME%\bin\catalina \
    run %1 %2 %3 %4 %5 %6 %7 %8 %9
set CATALINA_HOME=%_CATALINA_HOME%
set _CATALINA_HOME=
set JAVA_HOME=%_JAVA_HOME%
set JSSE_HOME=%_JSSE_HOME%
```

## 5. Erweiterung des ClassLoader

```
... set CATALINA_OPTS=-server -ms32m -mx64m  
-Dcatalina.config=file:../conf/catalina.properties ...
```

Um die ClassLoader des Tomcats zu erweitern, besteht seit dem Tomcat 5-Release die Möglichkeit die Steuerung durch ein *catalina.properties*-Datei. Hierzu muss der Tomcat mit einem zusätzlichen Parameter *catalina.config* gestartet werden. Dies erreicht man durch einen Eintrag in den *CATALINA\_OPTS* Variable in den Startup-Skripten.

### Note:

Man beachte das der Parameter *catalina.config* eine echte URL erwartet. Also Protokoll nicht vergessen (Beispiel: *file:../conf/catalina.properties*)

### Beispiel einer *catalina.properties*

```
package.access=sun.,org.apache.catalina.,org.apache.coyote., \  
org.apache.tomcat.,org.apache.jasper.  
package.definition=sun.,java.,org.apache.catalina.,org.apache.coyote. \  
org.apache.tomcat.,org.apache.jasper.  
common.loader=${catalina.base}/common/classes, \  
${catalina.base}/common/lib/*.jar, \  
${catalina.home}/common/classes, \  
${catalina.home}/common/endorsed/*.jar, \  
${catalina.home}/common/lib/*.jar  
server.loader=${catalina.base}/server/classes; \  
${catalina.home}/server/classes,${catalina.home}/server/lib/*.jar  
shared.loader=${catalina.base}/shared/classes,${catalina.base}/shared/lib/*.jar
```

### Note:

**WICHTIG:**Es handelt sich um eine Java Properties Datei und somit muss jeder Eintrag genau in einer Zeile stehen.

### Note:

**Tipp:**Für eigene Erweiterung wie JDBC Treiber, cgi-Servlet.jar, log4j oder JOTM dient das *common/lib* Verzeichnis

### Note:

**Tipp:**Es können in dieser Datei auch beliebige andere System Properties gesetzt werden.

### Note:

**Tipp:**Nur im Zusammenhang mit *\${catalina.home}* und *\${catalina.base}* ist eine automatisch Variablen-Expansion mit dem *\*.jar*Muster möglich.

**Note:**

**Tipp:** Seit dem Tomcat 5.0.19 zeigt der *shared.loader*, nun auf das *\${catalina.base}*. Somit ist es jetzt leichter Klassen bzw. jar-Archive zwischen allen Anwendung zu teilen und damit etwas Speicherplatz für gemeinsame Anwendung-Libs zu sparen. Bekanntlich laden Web-Container jeweils einzeln alle Libs im *WEB-INF/lib* Verzeichnis. **Achten** Sie darauf, dass diese Libs keine Singletons enthalten, die nur einmal pro Anwendung Sinn machen (log4J:-).

**Note:**

**Tipp:** Für den Betrieb unter Suse Linux kann das Skript *bin/tomcat-service.sh* zur Systemdienst Integration des Tomcats genutzt werden (s. README)

## 6. Ant Skripte

Gestartet wird mit dem Befehl *ant build.xml*. Mit einem Ant Skript kann man etwas mehr Luxus in seinen Tomcat bekommen und dann mit Leichtigkeit in Eclipse, Idea, JBuilder, Emacs oder JEdit integrieren. Einfach das Ant Skript anmelden und *zack* startet, *stop*, *secure* oder *debug* starten. Es gibt wirklich nichts leichteres als Java Programme in eine IDE via Ant zu integrieren.

Befehl	Bedeutung
tomstart	Start des Tomcats des Skripts <i>startup.bat</i> oder <i>startup.sh</i> je nach OS.
tomdebug	Start des Tomcats des Skripts <i>debug.bat</i> oder <i>debug.sh</i> je nach OS mit Debugger.
tomsecure	Start des Tomcats des Skripts <i>startup.bat</i> oder <i>startup.sh</i> je nach OS mit SecurityManager.
serverkey	Generiert einen Demo-Keystore für diesen Tomcat in die Datei <i>conf/catalina.keystore</i> .
browser	Ein Browser wird mit der Root Anwendung des Template gestartet. :-) Geht auf unter Linux mit dem Mozilla! Der Server läuft unter dem Port 7380.

## 7. Patch der Umgebung

Mit dem Befehl *ant -f patch.xml patch* wird aus dem Verzeichnis *templates* die Anpassung an eine konkrete webdev-server Umgebung erzeugt. Schauen Sie sich die Templates einfach genau an und bauen sich Ihre eigene Tomcat-Welt.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<!--
* Copyright @ 2004 Peter Roßbach (pr@objektpark.de)
*
* Not Warranty to use it.
-->
<project name="webdev server plus patch" default="patch" basedir=".">

  <property name="deploytarget" value=""/>

  <target name="patch">
    <property name="patchfile" value="build${deploytarget}.patch"/>
    <filter filtersfile="${patchfile}"/>

    <copy todir="bin" filtering="true" overwrite="true">
      <fileset dir="template/bin">
        <include name="*.bat"/>
        <include name="*.sh"/>
      </fileset>
    </copy>
    <fixcrlf srcdir="bin" includes="*.sh" eol="lf"/>
    <fixcrlf srcdir="bin" includes="*.bat" eol="crlf"/>
    <chmod dir="bin" includes="*.sh" perm="+x"/>
    <chmod dir="bin" type="dir" perm="750"/>

    <copy todir="." filtering="true" overwrite="true">
      <fileset dir="template">
        <include name="*.properties"/>
      </fileset>
    </copy>
    <property file="build.properties"/>
    <copy todir="conf/Catalina/${engine.host}" filtering="true" overwrite="true">
      <fileset dir="template/webapps">
        <include name="*.xml"/>
      </fileset>
    </copy>

    <copy todir="conf" filtering="true" overwrite="true">
      <fileset dir="template/conf">
        <include name="*.xml"/>
        <include name="*.properties"/>
        <include name="*.conf"/>
      </fileset>
    </copy>
    <chmod dir="conf" type="dir" includes="*" perm="700"/>
    <chmod perm="600" type="file" >
      <fileset dir="conf">
        <include name="**/*.xml"/>
        <include name="*.policy"/>
        <include name="*.properties"/>
      </fileset>
    </chmod>

    <mkdir dir="temp"/>
  </target>
</project>
```

```

<mkdir dir="logs"/>
<mkdir dir="work"/>
<chmod dir="template" type="dir" includes="*" perm="750"/>
<chmod dir="template" type="file" includes="*" perm="640"/>
<chmod dir="common" type="dir" includes="*" perm="750"/>
<chmod dir="common" type="file" includes="*" perm="640"/>
<chmod dir="temp" type="dir" perm="750"/>
<chmod dir="temp" type="dir" perm="750"/>
<chmod dir="logs" type="dir" perm="750"/>
<chmod dir="work" type="dir" perm="750"/>
<chmod dir="webapps" type="dir" includes="*" perm="750"/>
<chmod dir="webapps" type="file" includes="*" perm="640"/>
<chmod type="file" perm="640">
  <fileset dir=".">
    <include name="*.xml"/>
    <include name="*.patch"/>
    <include name="*.properties"/>
  </fileset>
</chmod>
</target>

<target name="clean">
  <property file="build.properties"/>
  <delete dir="bin"/>
  <delete dir="temp"/>
  <delete dir="logs"/>
  <delete dir="work"/>
  <delete dir="conf/Catalina/${engine.host}"/>
  <delete file="build.properties"/>
</target>
</project>

```

## 8. server.xml

Die Konfiguration des webdev-server im TomC@\_webdevplus\_XX stellt eine Entwicklerversion dar. Meist reicht dieser Minimalserver mit einem HTTP Connector, Realm, Logger und optional hinzuschaltbaren SSL Connector völlig aus. Einige Werte können über die zentrale Patch-Konfiguration in der Datei *build.patch* noch angepaßt werden.

```

<Server port="@server.shutdown.port@"
        shutdown="@server.shutdown.key@" debug="@server.debug@">

  <!-- Enable JMX MBeans support -->

  <Listener
    className="org.apache.catalina.mbeans.ServerLifecycleListener"/>
  <Listener
    className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener"/>

```

```
<!-- Global JNDI resources -->
<GlobalNamingResources>

  <!-- Editable user database that can also be used by
    UserDatabaseRealm to authenticate users -->
  <Resource name="UserDatabase" auth="Container"
    type="org.apache.catalina.UserDatabase"
    description="User database that can be updated and saved">
  </Resource>
  <ResourceParams name="UserDatabase">
    <parameter>
      <name>factory</name>
      <value>org.apache.catalina.users.MemoryUserDatabaseFactory
      </value>
    </parameter>
    <parameter>
      <name>pathname</name>
      <value>conf/tomcat-users.xml</value>
    </parameter>
  </ResourceParams>

</GlobalNamingResources>

<Service name="Catalina">

  <Connector port="@connector.http.port@"
    redirectPort="@connector.https.port@"/>
  <!--
  <Connector port="@connector.ajp.port@"
    redirectPort="@connector.https.port@" protocol="AJP1.3"/>
  <Connector port="@connector.https.port@"
    secure="true"
    scheme="https"
    acceptCount="100"
    clientAuth="false"
    sslProtocol="TLS"
    keystoreFile="../conf/catalina.keystore"
    keystorePass="changeit"/>
  -->
  <Engine name="Catalina" defaultHost="@engine.host@" >

    <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
      debug="0" resourceName="UserDatabase"/>

    <!-- Global logger unless overridden at lower levels -->
    <Logger className="org.apache.catalina.logger.FileLogger"
      prefix="catalina_log." suffix=".txt"
      timestamp="true"/>
    <!-- Developer Mode -->
    <Host
      name="@engine.host@"
      appBase="webapps"
      unpackWARs="false"
```

```

        autoDeploy="true"
        deployXML="true"
        deployOnStartup="true"
    >
    <DefaultContext reloadable="true" />
    <Logger
        className="org.apache.catalina.logger.FileLogger"
        prefix="host_@engine.host@_log." suffix=".txt"
        timestamp="true"/>
    <Valve className="org.apache.catalina.valves.AccessLogValve"
        prefix="access_@engine.host@_log-"
        suffix=".txt"
        pattern="common"
        resolveHosts="false"/>
    </Host>
</Engine>

</Service>

</Server>

```

## 9. Die Saat

Einen Tomcat an einer beliebiger Stelle einfach zu installieren, um mal kurz etwas auszuprobieren, wäre wirklich eine Bereicherung. Ohh, wie haben Lars, Michael und ich das beim Schreiben der TomC@ Kolumne schon vermisst. Bisher gelingt das nur wenigen Experten und von kurz, kann oftmals nicht die ernsthaft die Rede sein. Ich habe mir nun das Apache Forrest Projekt zum Vorbild genommen und nun das ersehnte Template für die Tomcat Saat (seed) vorbereitet.

Ein Ant-Skript in der Kombination mit ein bisschen Windows *cmd-Shell* oder UNIX *sh* Akrobatik und schon wird der Traum erfreuliche Realität. Im **webdevPLUS** befindet sich im Verzeichnis *TomC@\_webdevplus\_XXX/seed* die notwendige Mechanik für die zentrale Installation (Saatgut :-). Das Template wird im Verzeichnis *\$seed.home* installiert Die Angabe diese Verzeichnis befindet sich in der Datei *TomC@\_webdevplus\_xx/build.patch*

### 9.1. Installation

#### Note:

Achten Sie darauf, das in der *build.patch* Datei die Eigenschaft **seed.home** korrekt gesetzt ist und Sie mit dieser Datei auch den Installationstest gemacht haben. Sonst müssen Sie später von Hand z.B Ihre *buildLINUX.patch* in die Datei *\$seed.home/webdev-server/build.patch* kopieren.

```

edit build.patch
# Prüfen in build.patch der Parameter insbesondere seed.home= %SEED_HOME%
ant -Ddeploytarget=WIN patch

```

## TomC@ - HowTo's

```
# Kontrolle ob das Template wirklich ein gutes Vorbild ist und die build.patch den rich
cd seed
ant -f seedinstall.xml
# System Pfad anpassen
set PATH=%SEED_HOME%;%PATH%
```

Im Verzeichnis `$SEED_HOME` sollte sich nun ein *bin*- und *webdev-server*-Verzeichnis befinden. Für Anpassungen auf Ihre Maschine können Sie nun diese *build.patch* Datei anpassen.

### 9.2. Anwendung

```
mkdir MY_NEW_PROJEKT_SERVER
cd MY_NEW_PROJEKT_SERVER
seedtom
cd webdev-server
ant tomstart
```

#### Was ist passiert ?

Das Template in `$seed.home/webdev-server` wurde in das eigene Verzeichnis kopiert und dann direkt mit dem Befehl *ant patch* angepaßt. Dieser Befehl bewirkt das der neue Tomcat webdev-server an die allgemeine Einstellung Ihres Rechner angepaßt wird und damit sofort einsatzfähig ist. Damit dies problemlos klappt, muss die Datei `$seed.home/webdev-server/build.patch` auf dem korrekten Stand sein.

#### Note:

**WICHTIG:**Achten Sie darauf das im Verzeichnis `$seed.home` wirklich in Ihrem `$PATH` enthalten ist (which seedtom).

#### Note:

**WICHTIG:**Eine weitere Anwendung von *seedtom* wendet das Template immer wieder gnadenlos auf Ihr Projekt an!

#### Note:

**Tipp:**Es ist im Verzeichnis `$seed.home` alles vorbereitet, um weitere Templates hinterlegen zu können. Eine Kopie des *seedtom*-Skripts und Schreiben einer eignen *seed.xml* Datei und fertig ist Ihre erstes eigenes Saatgut.

#### Note:

**VORSICHT:**In der Datei `$seed.home/webdev-server/build.patch` können Sie nun Ihre Rechneranpassung durchführen und diese auf Ihre Projekte erneut überschreibend anwenden.

## 10. Tomcat als Systemdienst

Klar, die richtige Entscheidung einen Tomcat als Service zu betreiben ist sicherlich die Nutzung eines Service Wrappers. In der Tomcat Distribution steht dafür der Jakarta Commons Daemon bereit. Ein einfaches Skript reicht oftmals unter Linux völlig aus. Solch ein Service-Skript ist unter *webdev-server/template/bin/tomcatservice.sh* zu finden. Es ist unter Suse Linux 9.0 erfolgreich gelaufen.

Die Einstellung des Service Namens und Nutzers geschieht direkt durch die übliche *buildLINUX.patch* Datei.

```
linux.service.script=webdevplus
linux.service.user=tomdev
```

Ändern und nochmals mit *ant -Ddeploytarget=LINUX patch* anwenden.

```
cd webdev-server/bin
# Root Passwort erforderlich
sudo ./tomcatservice.sh install
```

Nun befindet sich unter */etc/init.d* das Service Skript *webdevplus* und wird nun bei jedem Neustart sofort gestartet.

Befehle:

```
# Start des Dienst
sudo /etc/init.d/webdevplus start
# Stop des Dienst
sudo /etc/init.d/webdevplus stop
# Restart des Dienst
sudo /etc/init.d/webdevplus restart
# uninstall des Dienst
sudo /etc/init.d/webdevplus uninstall
```

#### Note:

**Anmerkungen:** Noch ist der Dienst nicht ausreichend getestet und für einen stabilen Betrieb oder der Nutzung von Ports < 1024 wird der Einsatz des Java Service Wrapper von mir empfohlen.

Mit iptables läßt sich allerdings das Port Aufgabe elegant überwinden.

```
iptables -t nat -A OUTPUT -d localhost -p tcp --dport 80 -j REDIRECT --to-ports 7380
iptables -t nat -A OUTPUT -d YOUR HOSTNAME -p tcp --dport 80 -j REDIRECT --to-ports 7380
```

```
iptables -t nat -A PREROUTING -d YOUR HOSTNAME -p tcp --dport 80 -j REDIRECT --to-ports 7380
```

Weitere Information unter

<http://www.klawitter.de/tomcat80.html>

<http://linux.org.mt/article/tomcat-ports>

Im Connector der *webdev-server/template/conf/server.xml* muss für einen solche Weiterleitung der Proxy-Port gesetzt werden  
`<Connector port="7380 proxyPort="80" redirectPort="7443" />`

Hier der aktuelle Stand des *tomcatservice.sh* Skripts. Anregung und Verbesserungen sind mir sehr willkommen.

```
#!/bin/sh
```

## TomC@ - HowTo's

```
# Copyright @ 2004 Peter Roßbach (pr@objektpark.de)
#
# Source is only for non commercial usage.
# Source code is not allow to used at coachings material.
#
# Not Warranty to use it.
# -----
# start Script for the CATALINA webdev Server
# author Peter Rossbach (pr@objektpark.de)
#
### BEGIN INIT INFO
# Provides:                               @linux.service.script@
# Required-Start:   $local_fs $remote_fs $network $apache $httpd $apache2 $httpd2
# X-UnitedLinux-Should-Start: $named $time $apache $httpd $apache2 $httpd2
# Required-Stop:     $local_fs $remote_fs $network
# X-UnitedLinux-Should-Stop:
# Default-Start:     3 5
# Default-Stop:      0 1 2 6
# Short-Description: Peter Rossbach webdevPLUS
# Description:       Starts/Stops the Peter Rossbach webdevPLUS
### END INIT INFO

test -s /etc/rc.status && . /etc/rc.status

mode=$1

case "$mode" in
    'start')
        # Start webdevPLUS
        checkproc @catalina.basefullpath@/bin/startup.sh && \
            echo -n "Starting service @linux.service.script@" && \
            rc_status -v && rc_exit
        echo -n "Starting service @linux.service.script@"
        chown -R @linux.service.user@ @catalina.basefullpath@
        su - @linux.service.user@ -c '@catalina.basefullpath@/bin/startup.sh'
        rc_status -v
        ;;
    'stop')
        # Stop webdevPLUS
        echo -n "Shutting down service @linux.service.script@: "
        su - @linux.service.user@ -c '@catalina.basefullpath@/bin/shutdown.sh'
        rc_status -v
        ;;
    'restart')
        #Restarts the webdevPLUS
        $0 stop
        $0 start
        rc_status
        ;;
    'install')
        #installs this script as init-script
        cp @catalina.basefullpath@/bin/tomcatservice.sh /etc/rc.d/@linux.servic
```

```
        /sbin/chkconfig -a @linux.service.scrip@
        exit 0
    ;;

    'uninstall')
        #delete the init-script
        $0 stop
        /sbin/chkconfig -d @linux.service.scrip@
        rm -f /etc/rc.d/@linux.service.scrip@
        exit 0
    ;;
*)
    # usage
    cat >&2 <<-EOF
        Usage: $0 <command>

        where <command> is one of:
        start      - starts the webdevPLUS @linux.service.scrip@ Server
        stop       - stops the webdevPLUS @linux.service.scrip@ Server
        restart    - restarts the webdevPLUS @linux.service.scrip@ Server
        install    - installs this script as webdevPLUS @linux.service.scrip@
        uninstall - uninstall the webdevPLUS @linux.service.scrip@ Init-Script
        help      - prints this screen
    EOF
    exit 1
    ;;
esac
rc_exit
```