

Spis treści

Wdrożenie.....	3
Pobranie pakietu.....	3
Zgodność serwera z wersjami JDK i API.....	5
Pierwsze uruchomienie. Startowanie, zatrzymywanie serwera i podgląd bieżących logów.....	5
Wykorzystanie skryptu catalina.sh.....	8
Instalacja najnowszego Oracle JDK.....	8
Ustawianie zmiennych środowiskowych JAVA_HOME i PATH.....	9
Uruchamianie serwera z przekierowaniem logów do pliku dziennika.....	11
Zatrzymanie serwera z użyciem skryptu catalina.sh.....	11
Uruchomienie serwera z przekierowaniem logów na konsolę.....	12
Sprawdzanie wersji serwera Tomcat oraz wersji używanego przez niego JDK.....	13
Opcje uruchomieniowe dla skryptu CATALINA.....	14
JAVA_OPTS.....	14
CATALINA_OPTS.....	14
Parametry XMS, XMX i XX:MaxPermSize.....	15
CATALINA_OUT.....	15
CATALINA_TMPDIR.....	15
JAVA_HOME.....	15
Automatyczne uruchamianie Tomcata wraz ze startem systemu oraz dostęp do konsoli Web przez sieć.....	16
Zmienne środowiskowe związane z serwerem Tomcat.....	19
Zmiana portu nasłuchu.....	20
Aplikacja WEB do zarządzania serwerem – Tomcat Manager.....	22
Dostęp do aplikacji Tomcat Manager.....	22
Konfiguracja uprawnień dostępu do panelu zarządzania serwerem Tomcat.....	23
Sprawdzanie statusu serwera.....	26
Panel zarządzania aplikacjami.....	27
Wdrażanie aplikacji na serwer.....	29
Wdrażanie pliku WAR z użyciem Tomcat Managera.....	29
Automatyczne rozpakowywanie archiwum WAR.....	30
Wdrażanie bez użycia aplikacji Tomcat Manager.....	31
Deinstalacja aplikacji.....	32
Zmiana adresu aplikacji.....	34
Wdrażanie jako domyślna główna aplikacja.....	35
Zmiana adresu aplikacji wdrożonej jako archiwum WAR bez rozpakowywania.....	36
Zatrzymywanie i startowanie aplikacji z użyciem Tomcat Manager'a.....	37
Sesje aplikacji i ich rozłączanie.....	39
Konsola tekstowa i wykorzystanie skryptów do zarządzania serwerem.....	41
Uprawnienia.....	41
Zatrzymywanie i uruchamianie aplikacji z poziomu konsoli / skryptu.....	41
Lista wdrożonych aplikacji wraz z informacją o ilości sesji i stanie uruchomienia z poziomu konsoli.....	42
Przeładowanie aplikacji z poziomu konsoli.....	43
Sprawdzanie statusu serwera w trybie tekstowym.....	44
Sprawdzanie w trybie tekstowym sesji wskazanej aplikacji.....	44
Odinstalowywanie aplikacji z poziomu konsoli.....	45
Składowe serwera.....	46
Katalogi serwera.....	46

Najważniejsze pliki i ich zawartość.....	46
Pliki konfiguracyjne aplikacji.....	47
Konfiguracja serwera Tomcat.....	48
SERVER.XML.....	48
Element Server.....	49
Element Service.....	49
Element Connector.....	49
Element Engine.....	50
Element Host.....	51
Element Context.....	53
Debugowanie serwera i rozwiązywanie problemów.....	55
Rodzaje logów serwera Tomcat.....	55
Plik o nazwie catalina.out.....	55
Pliki o formacie catalina.yyyy-mm-dd.txt.....	55
Pliki o formacie localhost.yyyy-mm-dd.txt.....	55
Pliki o formacie localhost_access.log.yyyy-mm-dd.txt.....	56
Pliki o formacie manager.yyyy-mm-dd.txt.....	56
Szukanie przyczyn problemów.....	57

Wdrożenie

Pobranie pakietu

Wchodzimy na witrynę <http://www.tomcat.apache.org> i pobieramy odpowiadającą nam wersję. Ja pobieram najnowszą na dany moment. Wersje zip i tar.gz to spakowany serwer Tomcat, którego instalacja będzie się sprowadzała w zasadzie tylko do odpakowania go i nadania odpowiednich uprawnień. Ta wersja działa zarówno dla systemu Windows jak i Linux. Foldery instalacyjne zawierają zarówno skrypty .bat jak i .sh. Istnieje też możliwość pobrania wersji instalacyjnej dla systemu Windows. Niestety nie ma gotowych pakietów RPM lub DEB.

8.5.9

Please see the [README](#) file for packaging information. It explains what ev

Binary Distributions

- Core:
 - [zip](#) ([pgp](#), [md5](#), [sha1](#))
 - [tar.gz](#) ([pgp](#), [md5](#), [sha1](#))
 - [32-bit Windows zip](#) ([pgp](#), [md5](#), [sha1](#))
 - [64-bit Windows zip](#) ([pgp](#), [md5](#), [sha1](#))
 - [32-bit/64-bit Windows Service Installer](#) ([pgp](#), [md5](#), [sha1](#))
- Full documentation:

Wybrałem wersję ZIP, pobrałem i rozpakowałem w katalogu w którym Tomcat będzie się docelowo znajdował. Po rozpakowaniu powinniśmy zobaczyć podkatalogi bin,conf, lib, logs, temp, webapps i work.

```
[andrew@localhost apache-tomcat-8.5.9]$ pwd
/home/andrew/soft/apache-tomcat-8.5.9
[andrew@localhost apache-tomcat-8.5.9]$ ls -la
total 128
drwxr-xr-x. 9 andrew andrew 4096 Dec  5 20:20 .
drwxrwxr-x. 3 andrew andrew 4096 Jan 11 19:36 ..
drwxr-xr-x. 2 andrew andrew 4096 Dec  5 20:19 bin
drwxr-xr-x. 2 andrew andrew 4096 Dec  5 20:19 conf
drwxr-xr-x. 2 andrew andrew 4096 Dec  5 20:19 lib
-rw-r--r--. 1 andrew andrew 58153 Dec  5 20:19 LICENSE
drwxr-xr-x. 2 andrew andrew 4096 Dec  5 20:18 logs
-rw-r--r--. 1 andrew andrew 1774 Dec  5 20:19 NOTICE
-rw-r--r--. 1 andrew andrew 7240 Dec  5 20:19 RELEASE-NOTES
-rw-r--r--. 1 andrew andrew 16416 Dec  5 20:19 RUNNING.txt
drwxr-xr-x. 2 andrew andrew 4096 Dec  5 20:19 temp
drwxr-xr-x. 7 andrew andrew 4096 Dec  5 20:19 webapps
drwxr-xr-x. 2 andrew andrew 4096 Dec  5 20:18 work
[andrew@localhost apache-tomcat-8.5.9]$
```

W systemie Linux musimy jeszcze dodać uprawnienia do uruchamiania skryptów z podkatalogu bin – to tutaj znajdują się wszystkie niezbędne do uruchamiania, zatrzymywania i konfiguracji Tomcata skrypty.

```
[andrew@localhost apache-tomcat-8.5.9]$ chmod 777 bin/*
```

Zgodność serwera z wersjami JDK i API

Wersja Tomcata	Servlet API	JSP API	JDK
3.0	2.2	1.1	1.1
4.1	2.3	1.2	1.3
5.5	2.4	2.0	1.4
6.0	2.5	2.1	1.5
7.0	3.0	2.2	1.6
8.0	3.1	2.3	1.7

Pierwsze uruchomienie. Startowanie, zatrzymywanie serwera i podgląd bieżących logów

Aby uruchomić serwer Tomcat należy wykonać skrypt **startup.sh** (lub **startup.bat** w systemie Windows) znajdujący się w podkatalogi BIN. Z użyciem narzędzia **tail** możemy podejrzeć logi serwera aby sprawdzić czy serwer uruchomił się poprawnie.

Wystarczy wywołać **tail -f logs/catalina.out**:

```
[andrew@localhost apache-tomcat-8.5.9]$ bin/startup.sh
Using CATALINA_BASE:   /home/andrew/soft/apache-tomcat-8.5.9
Using CATALINA_HOME:   /home/andrew/soft/apache-tomcat-8.5.9
Using CATALINA_TMPDIR: /home/andrew/soft/apache-tomcat-8.5.9/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /home/andrew/soft/apache-tomcat-8.5.9/bin/bootstrap.jar:/home/andrew/soft/apache-tomcat-8.5.9/bin/tomcat-juli.jar
Tomcat started.
[andrew@localhost apache-tomcat-8.5.9]$ tail -f logs/catalina.out
11-Jan-2017 19:37:34.589 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web application directory /home/andrew/soft/apache-tomcat-8.5.9/webapps/host-manager has finished in 96 ms
11-Jan-2017 19:37:34.589 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploying web application directory /home/andrew/soft/apache-tomcat-8.5.9/webapps/docs
11-Jan-2017 19:37:34.628 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web application directory /home/andrew/soft/apache-tomcat-8.5.9/webapps/docs has finished in 39 ms
11-Jan-2017 19:37:34.632 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploying web application directory /home/andrew/soft/apache-tomcat-8.5.9/webapps/examples
11-Jan-2017 19:37:35.297 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web application directory /home/andrew/soft/apache-tomcat-8.5.9/webapps/examples has finished in 665 ms
11-Jan-2017 19:37:35.301 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploying web application directory /home/andrew/soft/apache-tomcat-8.5.9/webapps/manager
11-Jan-2017 19:37:35.366 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web application directory /home/andrew/soft/apache-tomcat-8.5.9/webapps/manager has finished in 65 ms
11-Jan-2017 19:37:35.375 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler [http-nio-8080]
11-Jan-2017 19:37:35.394 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler [ajp-nio-8009]
11-Jan-2017 19:37:35.397 INFO [main] org.apache.catalina.startup.Catalina.start Server startup in 1949 ms
```

W systemie Windows również możemy zainstalować windowsową adaptację programu Tail dostępną pod adresem <http://tailforwin32.sourceforge.net/> . W logach powinniśmy zobaczyć dwa wpisy takie jak zaznaczone na powyższej ilustracji. Pierwszy zaznaczony fragment odnosi się do portu na którym uruchomiony zostaje serwer Tomcat, domyślnie jest to port **8080**. Drugi to informacja o poprawnym wystartowaniu serwera.

Gdyby podczas uruchamiania okazało że start serwera nie powiódł się, może to być (i najczęściej jest) spowodowane zajętością któregoś z używanych przez Tomcata portów. Przede wszystkim sprawdź czy dostępny jest port 8080, jeśli tak to sprawdź czy możesz tymczasowo wyłączyć blokującą usługę, a jeśli nie to zmień port nasłuchu Tomcata na inny. Tomcat domyślnie potrzebuje też portu 8005. Możesz tego dokonać edytując plik **server.xml** znajdujący się w podkatalogu **conf**. Proces ten szczegółowo został opisany w jednym z kolejnych rozdziałów.

Aby zatrzymać serwer, wystarczy wywołać skrypt **shutdown.sh** znajdujący się w podkatalogu BIN.

```
[andrew@localhost apache-tomcat-8.5.9]$ bin/shutdown.sh
Using CATALINA_BASE:   /home/andrew/soft/apache-tomcat-8.5.9
Using CATALINA_HOME:   /home/andrew/soft/apache-tomcat-8.5.9
Using CATALINA_TMPDIR: /home/andrew/soft/apache-tomcat-8.5.9/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /home/andrew/soft/apache-tomcat-8.5.9/bin/bootstrap.jar:/home/andrew/soft/apache-tomcat-8.5.9/bin/tomcat-juli.jar
```

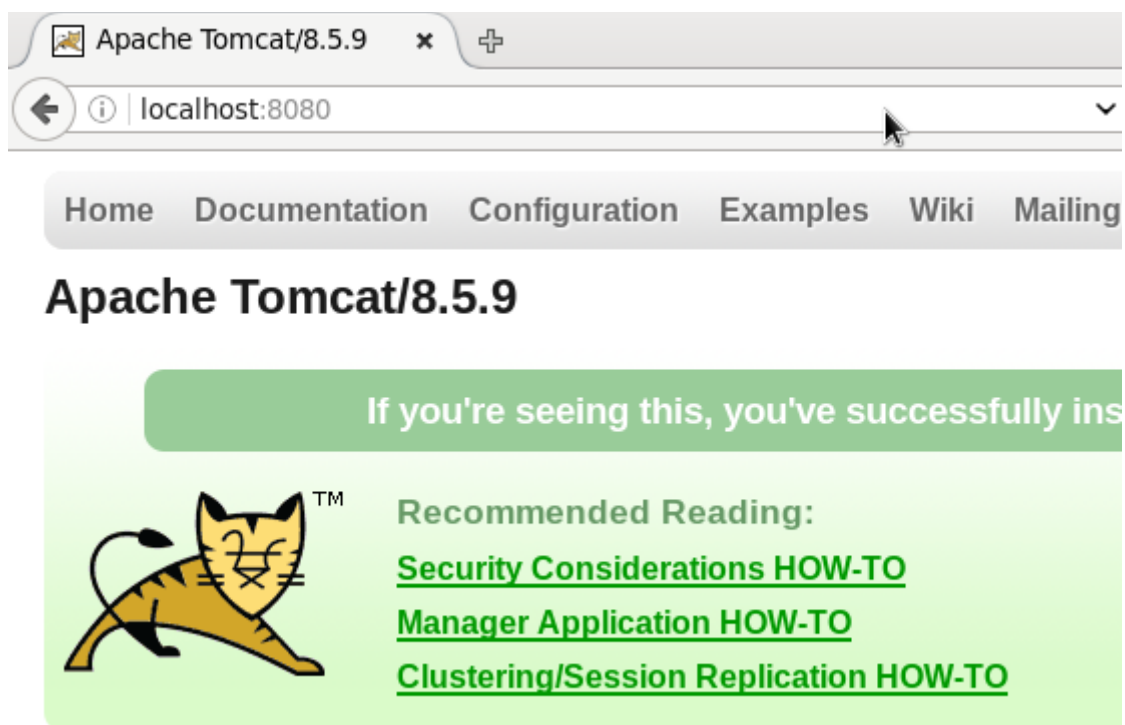
Po uruchomieniu serwera warto jeszcze sprawdzić czy faktycznie nasłuchuje na wskazanym porcie. Możemy zrobić to przy użyciu narzędzia NMAP w systemie Linux lub ZenMap w systemie Windows (<https://nmap.org/zenmap/>).

```
[andrew@localhost apache-tomcat-8.5.9]$ nmap localhost

Starting Nmap 5.51 ( http://nmap.org ) at 2017-01-11 19:40 CET
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0011s latency).
Other addresses for localhost (not scanned): 127.0.0.1
Not shown: 994 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
25/tcp    open  smtp
111/tcp   open  rpcbind
631/tcp   open  ipp
8009/tcp  open  ajp13
8080/tcp  open  http-proxy

Nmap done: 1 IP address (1 host up) scanned in 0.24 seconds
[andrew@localhost apache-tomcat-8.5.9]$
```

Jeśli wszystko jest w porządku wprowadź w przeglądarce adres <http://localhost:8080/> (lub inna wartość zamiast 8080 w przypadku zmiany portu nasłuchu). Powinieneś zobaczyć stronę startową Tomcata:



Wykorzystanie skryptu catalina.sh

Zamiast posługiwać się skryptami startup.sh i shutdown.sh możemy uruchamiać i zatrzymywać serwer Tomcat z użyciem skryptu catalina.sh również znajdującego się w podkatalogu bin.

Daje nam to dodatkową możliwość kierowania strumienia logów, sprawdzenia wersji Tomcata czy debuggowanie, a także uruchomienia Tomcata z alternatywnym plikiem konfiguracyjnym.

Aby skrypt ten działał poprawnie, potrzebujemy mieć zainstalowane JDK (najlepiej w najnowszej wersji) i ustawioną zmienną środowiskową **JAVA_HOME**. Poniżej podaję sposób instalacji pakietu Oracle JDK w wersji 8.

Instalacja najnowszego Oracle JDK

Przechodzimy do podkatalogu w którym ma się znaleźć plik instalacyjny JDK i z użyciem polecenia wget pobieramy odpowiedni pakiet:

```
wget --header "Cookie: oraclelicense=accept-securebackup-cookie"  
http://download.oracle.com/otn-pub/java/jdk/8u102-b14/jdk-8u102-linux-x64.rpm
```

Następnie instalujemy pakiet:

```
rpm -ivh jdk-8u102-linux-x64.rpm
```

Jeśli mamy zainstalowany inny pakiet Java (np. samo JRE) warto jeszcze przestawić domyślną Javę na tę właśnie zainstalowaną. Wydajemy poniższą komendę i wybieramy javę która nam odpowiada:

```
alternatives --config java
```

```
[root@localhost soft]# alternatives --config java  
There are 3 programs which provide 'java'.  
  
  Selection    Command  
-----  
  1            /usr/lib/jvm/jre-1.7.0-openjdk.x86_64/bin/java  
  2            /usr/lib/jvm/jre-1.6.0-openjdk.x86_64/bin/java  
*+ 3          /usr/java/jdk1.8.0_102/jre/bin/java  
Enter to keep the current selection[+], or type selection number: 3
```


Ustawianie zmiennych środowiskowych JAVA_HOME i PATH

Sprawdźmy na początku czy aby zmiennej JAVA_HOME nie mamy już ustawionej, oraz czy ścieżka do katalogu JDK nie jest już dodana do zmiennej PATH. Aby sprawdzić zmienną JAVA_HOME wystarczy wywołać polecenie:

echo \$JAVA_HOME

```
[root@localhost soft]# echo $JAVA_HOME  
[root@localhost soft]#
```

Jeśli zmienna jest pusta, musimy ją ustawić.

Aby sprawdzić zawartość zmiennej PATH wywołujemy analogiczne polecenie:

echo \$PATH

```
[root@localhost soft]# echo $PATH  
/usr/lib64/qt-3.3/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin  
[root@localhost soft]#
```

Ustawiamy obie zmienne w ramach naszej konsoli:

```
export JAVA_HOME=/usr/java/jdk1.8.0_102/  
export PATH=$PATH:/usr/java/jdk1.8.0_102/bin
```

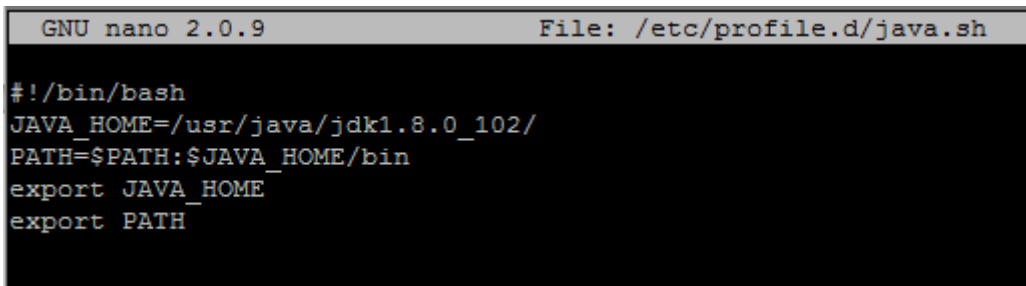
Upewniamy się czy zmienne faktycznie zostały ustawione:

```
[root@localhost soft]# echo $JAVA_HOME
/usr/java/jdk1.8.0_102/
[root@localhost soft]# echo $PATH
/usr/lib64/qt-3.3/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin:/usr/java
/jdk1.8.0_102/bin:/usr/java/jdk1.8.0_102/bin
[root@localhost soft]#
```

Ta metoda sprawi jednak że wszystko będzie działało tylko w ramach aktualnej sesji w aktualnej konsoli, warto więc zadbać o to by zmienne te były ustawiane również przy starcie systemu.

Aby zmienne ustawiały się również przy starcie systemu tworzymy plik **java.sh** w katalogu **/etc/profile.d** i uzupełniamy następującą treścią:

```
#!/bin/bash
JAVA_HOME=/usr/java/jdk1.8.0_102/
PATH=$PATH:$JAVA_HOME/bin
export JAVA_HOME
export PATH
```



```
GNU nano 2.0.9 File: /etc/profile.d/java.sh
#!/bin/bash
JAVA_HOME=/usr/java/jdk1.8.0_102/
PATH=$PATH:$JAVA_HOME/bin
export JAVA_HOME
export PATH
```

Nadajemy skryptowi uprawnienia uruchamiania:

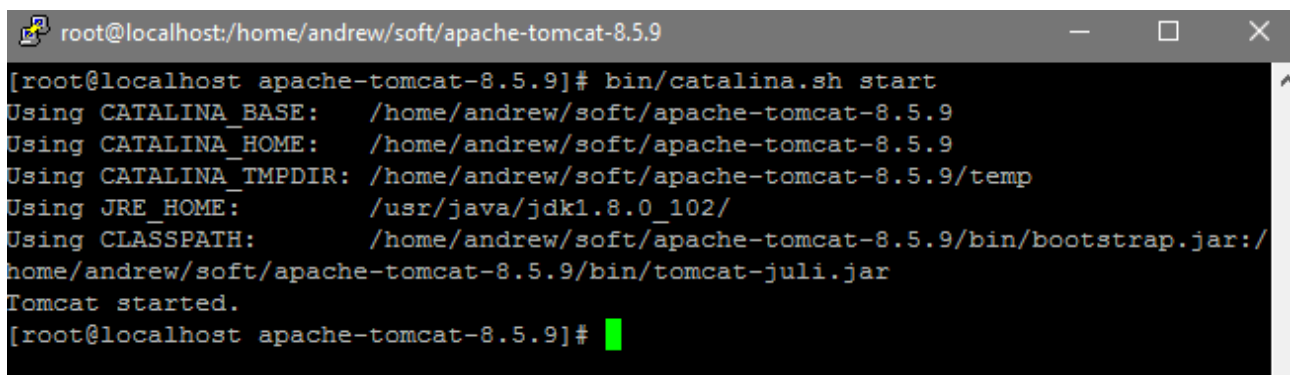
```
chmod +x /etc/profile.d/java.sh
```

Po uruchomieniu sprawdź czy na pewno zmienne środowiskowe są poprawnie ustawione.

Uruchamianie serwera z przekierowaniem logów do pliku dziennika

Wracając do obsługi skryptu catalina, możemy wywołać ten skrypt z przełącznikiem start co spowoduje standardowe uruchomienie serwera Tomcat z przekierowaniem logów do pliku dziennika catalina.out (znajdującego się w podkatalogu LIB). Serwer uruchomi się w dokładnie taki sam sposób jak w przypadku wywołania skryptu startup.sh

./catalina.sh start

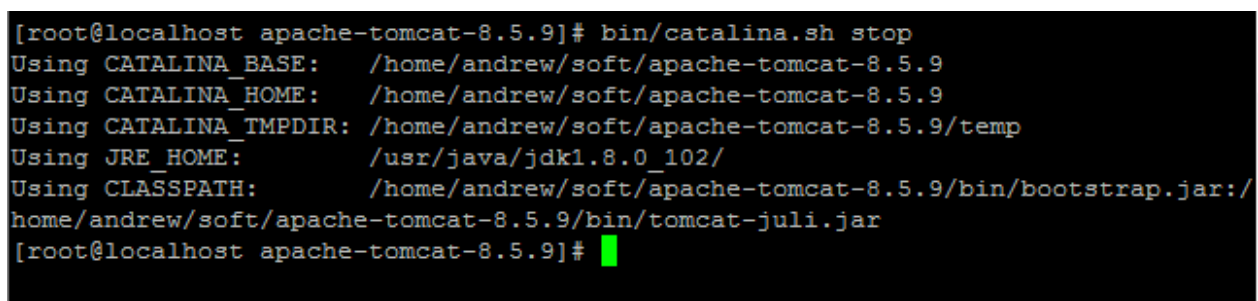
A terminal window with a dark background and light text. The title bar shows 'root@localhost:/home/andrew/soft/apache-tomcat-8.5.9'. The terminal content shows the command 'bin/catalina.sh start' being executed. The output lists environment variables: CATALINA_BASE, CATALINA_HOME, CATALINA_TMPDIR, JRE_HOME, and CLASSPATH. It then says 'Tomcat started.' and returns to the prompt. A green cursor is visible at the end of the prompt line.

```
root@localhost:/home/andrew/soft/apache-tomcat-8.5.9
[root@localhost apache-tomcat-8.5.9]# bin/catalina.sh start
Using CATALINA_BASE:   /home/andrew/soft/apache-tomcat-8.5.9
Using CATALINA_HOME:   /home/andrew/soft/apache-tomcat-8.5.9
Using CATALINA_TMPDIR: /home/andrew/soft/apache-tomcat-8.5.9/temp
Using JRE_HOME:        /usr/java/jdk1.8.0_102/
Using CLASSPATH:       /home/andrew/soft/apache-tomcat-8.5.9/bin/bootstrap.jar:/
home/andrew/soft/apache-tomcat-8.5.9/bin/tomcat-juli.jar
Tomcat started.
[root@localhost apache-tomcat-8.5.9]#
```

Zatrzymanie serwera z użyciem skryptu catalina.sh

Do zatrzymania serwera używamy przełącznika stop:

./catalina.sh stop

A terminal window with a dark background and light text. The title bar shows 'root@localhost apache-tomcat-8.5.9'. The terminal content shows the command 'bin/catalina.sh stop' being executed. The output lists environment variables: CATALINA_BASE, CATALINA_HOME, CATALINA_TMPDIR, JRE_HOME, and CLASSPATH. It then returns to the prompt. A green cursor is visible at the end of the prompt line.

```
[root@localhost apache-tomcat-8.5.9]# bin/catalina.sh stop
Using CATALINA_BASE:   /home/andrew/soft/apache-tomcat-8.5.9
Using CATALINA_HOME:   /home/andrew/soft/apache-tomcat-8.5.9
Using CATALINA_TMPDIR: /home/andrew/soft/apache-tomcat-8.5.9/temp
Using JRE_HOME:        /usr/java/jdk1.8.0_102/
Using CLASSPATH:       /home/andrew/soft/apache-tomcat-8.5.9/bin/bootstrap.jar:/
home/andrew/soft/apache-tomcat-8.5.9/bin/tomcat-juli.jar
[root@localhost apache-tomcat-8.5.9]#
```

Uruchomienie serwera z przekierowaniem logów na konsolę

Możemy także uruchomić skrypt catalina z przełącznikiem run:

```
./catalina.sh run
```

sprawi to że Tomcat zostanie uruchomiony, ale logi serwera zostaną przekierowane na konsolę z której uruchamialiśmy skrypt catalina. Jest to o tyle wygodne że nie musimy dodatkowo uruchamiać kolejnej komendy w celu podglądu logów, jednak jeśli przerwiemy działanie skryptu np. uruchamiając kombinację CTRL+C działanie serwera zostanie przerwane!

```
[root@localhost apache-tomcat-8.5.9]# bin/catalina.sh run
Using CATALINA_BASE:   /home/andrew/soft/apache-tomcat-8.5.9
Using CATALINA_HOME:   /home/andrew/soft/apache-tomcat-8.5.9
Using CATALINA_TMPDIR: /home/andrew/soft/apache-tomcat-8.5.9/temp
Using JRE_HOME:        /usr/java/jdk1.8.0_102/
Using CLASSPATH:       /home/andrew/soft/apache-tomcat-8.5.9/bin/bootstrap.jar:/home/andrew/soft/apache-tomcat-8.5.9/
12-Jan-2017 14:16:52.908 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server version:   Apache/2.4.18
12-Jan-2017 14:16:52.922 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server built:    Dec 17 2012 12:29:05
12-Jan-2017 14:16:52.922 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server number:   8.5.9
12-Jan-2017 14:16:52.922 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log OS Name:        Linux
12-Jan-2017 14:16:52.922 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log OS Version:    2.6.32-1.el6.elrepo.x86_64
12-Jan-2017 14:16:52.922 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Architecture: amd64
12-Jan-2017 14:16:52.922 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Java Home:     /usr/java/jdk1.8.0_102
12-Jan-2017 14:16:52.922 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log JVM Version:   1.8.0_102
12-Jan-2017 14:16:52.922 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log JVM Vendor:    Oracle Corporation
12-Jan-2017 14:16:52.922 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log CATALINA_BASE: /home/andrew/soft/apache-tomcat-8.5.9
12-Jan-2017 14:16:52.922 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log CATALINA_HOME: /home/andrew/soft/apache-tomcat-8.5.9
12-Jan-2017 14:16:52.924 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dj
12-Jan-2017 14:16:52.925 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dj
12-Jan-2017 14:16:52.925 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dj
12-Jan-2017 14:16:52.926 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dj
12-Jan-2017 14:16:52.926 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dc
12-Jan-2017 14:16:52.926 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dc
12-Jan-2017 14:16:52.927 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dj
12-Jan-2017 14:16:52.927 INFO [main] org.apache.catalina.core.AprLifecycleListener.lifecycleEvent The APR based Apache
  ch: /usr/java/packages/lib/amd64:/usr/lib64:/lib64:/lib:/usr/lib
12-Jan-2017 14:16:53.160 INFO [main] org.apache.coyote.AbstractProtocol.init Initializing ProtocolHandler ["http-nio-
12-Jan-2017 14:16:53.195 INFO [main] org.apache.tomcat.util.net.NioSelectorPool.getSharedSelector Using a shared sele
12-Jan-2017 14:16:53.206 INFO [main] org.apache.coyote.AbstractProtocol.init Initializing ProtocolHandler ["ajp-nio-8
12-Jan-2017 14:16:53.208 INFO [main] org.apache.tomcat.util.net.NioSelectorPool.getSharedSelector Using a shared sele
12-Jan-2017 14:16:53.209 INFO [main] org.apache.catalina.startup.Catalina.load Initialization processed in 1079 ms
```

Sprawdzanie wersji serwera Tomcat oraz wersji używanego przez niego JDK

Aby sprawdzić wersję serwera Tomcat wystarczy wywołać skrypt catalina z przełącznikiem version:

./catalina.sh version

```
[root@localhost apache-tomcat-8.5.9]# bin/catalina.sh version
Using CATALINA_BASE:   /home/andrew/soft/apache-tomcat-8.5.9
Using CATALINA_HOME:   /home/andrew/soft/apache-tomcat-8.5.9
Using CATALINA_TMPDIR: /home/andrew/soft/apache-tomcat-8.5.9/temp
Using JRE_HOME:        /usr/java/jdk1.8.0_102/
Using CLASSPATH:       /home/andrew/soft/apache-tomcat-8.5.9/bin
Server version: Apache Tomcat/8.5.9
Server built:   Dec 5 2016 20:18:12 UTC
Server number: 8.5.9.0
OS Name:        Linux
OS Version:     2.6.32-642.11.1.el6.x86_64
Architecture:  amd64
JVM Version:    1.8.0_102-b14
JVM Vendor:     Oracle Corporation
[root@localhost apache-tomcat-8.5.9]#
```

Opcje uruchomieniowe dla skrytu CATALINA

JAVA_OPTS

Zmienna środowiskowa JAVA_OPTS jest dostępna dla wszystkich procesów JAVA, włącznie z serwerem Tomcat. Można z jej użyciem ustawić np. kodowanie z użyciem którego będą czytane pliki:

```
export JAVA_OPTS="-Dfile.encoding=utf-8"
```

Jeśli po wyeksportowaniu tego ustawienia, z tej samej konsoli uruchomisz Tomcata, możesz być pewien że dla niego te ustawienia będą zastosowane. Możesz także zmodyfikować plik `/etc/profile.d/` tak by ustawienia te były stosowane zawsze, w tym dla serwera Tomcat uruchamianego automatycznie razem ze startem systemu. Pamiętaj jednak że to ustawienie dotyczy wszystkich procesów JAVA na danej maszynie.

CATALINA_OPTS

Jeśli chciałbyś ustawić jakieś opcje uruchomieniowe tylko dla serwera Tomcat, możesz posłużyć się zmienną środowiskową CATALINA_OPTS. Ta opcja jest najczęściej wykorzystywana do ustawienia dostępnej dla Tomcata pamięci operacyjnej co ustawia się w ten sposób:

```
export CATALINA_OPTS = „-Xms256m -Xmx1g -XX:MaxPermSize=265m”
```

Pamiętaj że jest to ustawienie dla aktualnej konsoli, natomiast ten parametr podobnie jak JAVA_OPTS możesz ustawić w `/etc/profile.d/`. Taki sam wpis możesz dodać także w pliku `catalina.sh` – zadziała identycznie.

Aby zmiany zadziałały, musisz zrestartować serwer Tomcat.

Parametry XMS, XMx i XX:MaxPermSize

Parametr **XMS** określa minimalną ilość pamięci która ma zostać zaalokowana dla wirtualnej maszyny Javy na potrzeby uruchamianych aplikacji – w tym serwera Tomcat. Pamiętaj że jeśli określisz wartość większą niż dostępna, serwer Tomcat nie uruchomi się. Domyślna wartość tego parametru to 64MB.

Parametr **XMx** określa maksymalną ilość pamięci jaką może zająć wirtualna maszyna Javy na potrzeby uruchamianych aplikacji – w tym serwera Tomcat. Domyślna wartość tego parametru to 64MB.

Parametr **XX:MaxPermSize** określa maksymalną dostępną pamięć na potrzeby samej Javy. Nie zawiera się w ramach XMS ani XMx – funkcjonuje niejako „obok”.

Jednostki podajemy z użyciem k,m lub g określającym odpowiednio kilobajty, megabajty i gigabajty. Wirtualna maszyna startując zaalokuje przestrzeń określoną parametrem XMS i będzie ją w razie potrzeby zwiększać aż do osiągnięcia wartości określanej przez XMx. Jeśli dysponujesz dużą ilością pamięci operacyjnej, możesz ustawić te dwa parametry na taką samą wartość. Dzięki temu nie będziemy wytracać czasu na alokowanie kolejnych fragmentów pamięci.

CATALINA_OUT

Tym parametrem możemy zmienić domyślne miejsce zrzucania logów. Jeśli tego nie zmienimy, wszystkie logi będą lądowały w logs/catalina.out

CATALINA_TMPDIR

Tym parametrem możemy zmienić lokalizację na tymczasowe pliki tworzone przez serwer Tomcat. Domyślnie jest ustawiony podkatalog temp serwera.

JAVA_HOME

Miejsce instalacji Javy.

Automatyczne uruchamianie Tomcata wraz ze startem systemu oraz dostęp do konsoli Web przez sieć.

Najprostszą metodą na zrealizowanie tego zadania będzie dodanie wpisu do skryptu `/etc/rc.d/rc.local` :

```
nano /etc/rc.d/rc.local ]
```

Robimy to oczywiście z poziomu użytkownika **root**. Na dole skryptu dodajemy ścieżkę do **startup.sh**:

```
GNU nano 2.0.9 File: /etc/rc.d/rc.local

#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here if you don't
# want to do the full Sys V style init stuff.

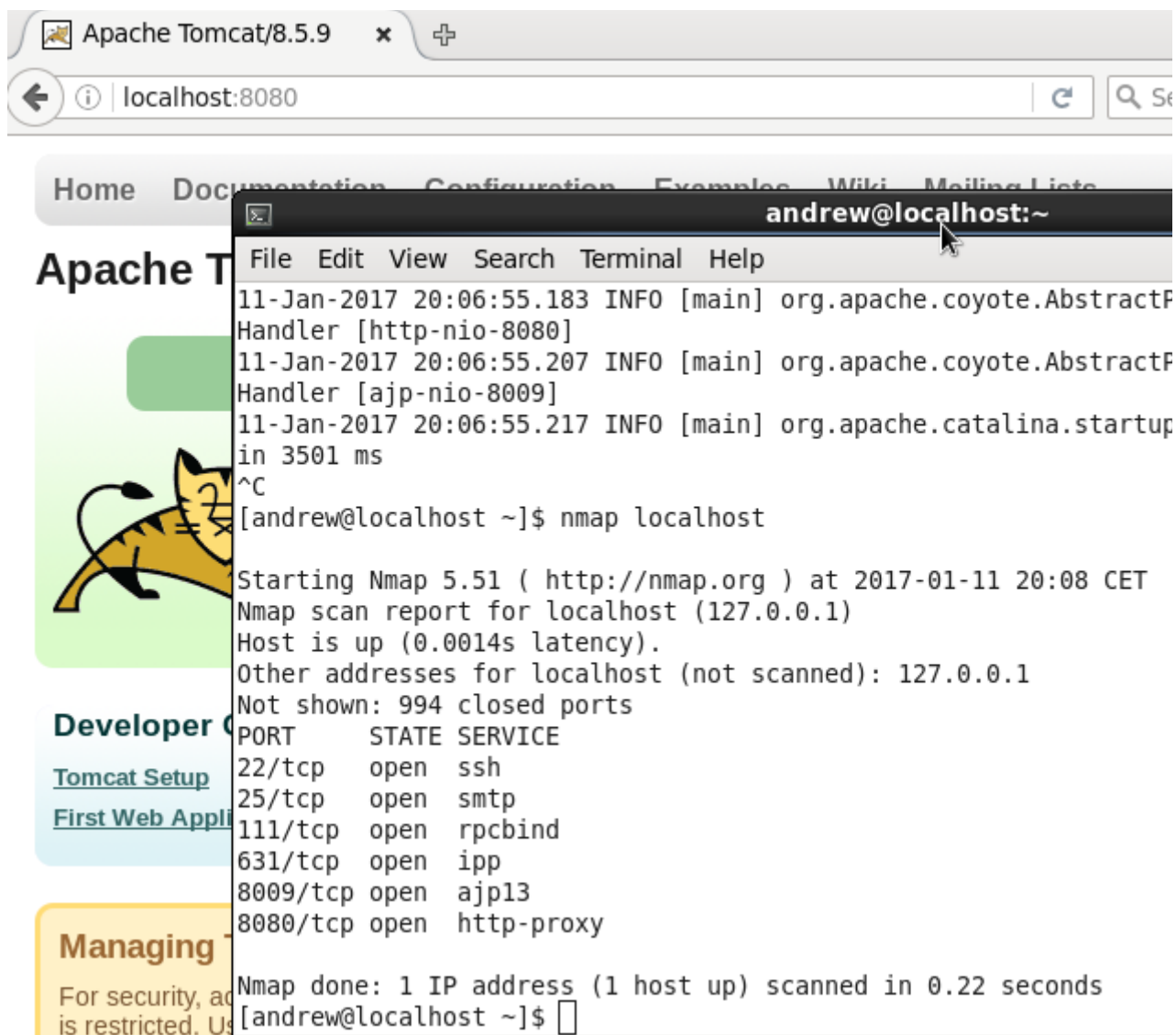
touch /var/lock/subsys/local
/home/andrew/soft/apache-tomcat-8.5.9/bin/startup.sh
]
```


Po wprowadzeniu i zapisaniu zmian restartujemy hosta i sprawdzamy logi Tomcata aby sprawdzić czy serwer faktycznie się uruchomił:

```
andrew@localhost:~  
File Edit View Search Terminal Help  
[andrew@localhost ~]$ tail -f /home/andrew/soft/apache-tomcat-8.5.9/logs/catalina.out
```

```
andrew@localhost:~  
File Edit View Search Terminal Help  
ployDirectory Deploying web application directory /home/andrew/soft/apache-tomcat-8.5.9/webapps/docs  
11-Jan-2017 20:06:54.286 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web application directory /home/andrew/soft/apache-tomcat-8.5.9/webapps/docs has finished in 25 ms  
11-Jan-2017 20:06:54.291 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploying web application directory /home/andrew/soft/apache-tomcat-8.5.9/webapps/examples  
11-Jan-2017 20:06:55.080 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web application directory /home/andrew/soft/apache-tomcat-8.5.9/webapps/examples has finished in 789 ms  
11-Jan-2017 20:06:55.085 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploying web application directory /home/andrew/soft/apache-tomcat-8.5.9/webapps/manager  
11-Jan-2017 20:06:55.171 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web application directory /home/andrew/soft/apache-tomcat-8.5.9/webapps/manager has finished in 86 ms  
11-Jan-2017 20:06:55.183 INFO [main] org.apache.coyote.AbstractProtocol.start Starting Protocol Handler [http-nio-8080]  
11-Jan-2017 20:06:55.207 INFO [main] org.apache.coyote.AbstractProtocol.start Starting Protocol Handler [ajp-nio-8009]  
11-Jan-2017 20:06:55.217 INFO [main] org.apache.catalina.startup.Catalina.start Server startup in 3501 ms
```

Dla pewności sprawdźmy jeszcze otwarte porty i dostęp poprzez przeglądarkę :



The screenshot shows a web browser window with the Apache Tomcat 8.5.9 homepage. The browser's address bar shows 'localhost:8080'. The page content includes navigation links (Home, Documentation, Configuration, Examples, Wiki, Mailing Lists), the Apache Tomcat logo, and sections for 'Developer Center' (with links for Tomcat Setup and First Web Application) and 'Managing' (with a note about security restrictions).

Overlaid on the browser is a terminal window titled 'andrew@localhost:~'. The terminal output shows the following:

```
File Edit View Search Terminal Help
11-Jan-2017 20:06:55.183 INFO [main] org.apache.coyote.AbstractF
Handler [http-nio-8080]
11-Jan-2017 20:06:55.207 INFO [main] org.apache.coyote.AbstractF
Handler [ajp-nio-8009]
11-Jan-2017 20:06:55.217 INFO [main] org.apache.catalina.startup
in 3501 ms
^C
[andrew@localhost ~]$ nmap localhost

Starting Nmap 5.51 ( http://nmap.org ) at 2017-01-11 20:08 CET
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0014s latency).
Other addresses for localhost (not scanned): 127.0.0.1
Not shown: 994 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
25/tcp    open  smtp
111/tcp   open  rpcbind
631/tcp   open  ipp
8009/tcp  open  ajp13
8080/tcp  open  http-proxy

Nmap done: 1 IP address (1 host up) scanned in 0.22 seconds
[andrew@localhost ~]$
```

Zmienne środowiskowe związane z serwerem Tomcat

CATALINA_HOME	Zmienna określająca miejsce instalacji Tomcata.
CATALINA_OPTS	Przekazuje javie opcje do uruchomienia Tomcata
CATALINA_TMPDIR	Katalog na pliki tymczasowe tomcata (domyślnie wskazuje na podkatalog temp w katalogu instalacji Tomcata).
JAVA_HOME	Wskazuje miejsce instalacji używanej przez Tomcata Javy
JAVA_OPTS	Umożliwia ustawienie zmiennych środowiska Java

Zmiana portu nasłuchu

Aby zmienić port nasłuchu serwera Tomcat z domyślnego 8080 na inny, wystarczy wyedytować plik `conf/server.xml` i odnaleźć sekcję widoczną poniżej:

```
<Connector port="8080" protocol="HTTP/1.1"
  connectionTimeout="20000"
  redirectPort="8443" />
```

następnie zmienić port na inny:

```
<Connector port="80" protocol="HTTP/1.1"
  connectionTimeout="20000"
  redirectPort="8443" />
```

Jeśli chcesz zmienić porty nasłuchu Tomcata ponieważ porty domyślnie przez niego alokowane są zajęte, pamiętaj też o zmianie portu 8005 który również jest domyślnie otwierany w celu umożliwienia wyłączenia Tomcata:

```
-->
<Server port="8005" shutdown="SHUTDOWN">
  <Listener className="org.apache.catalina.st
  <!-- Security listener. Documentation at /d
```

Plik server.xml jest czytany tylko przy starcie Tomcata, a to sprawia że aby nasze zmiany zostały utrwalone należy po edycji tego pliku zrestartować serwer.

bin/shutdown.sh

bin/startup.sh

Po zmianie możesz sprawdzić np. narzędziem nmap czy faktycznie otwarty został nowy port:

```
[root@localhost apache-tomcat-8.5.9]# nmap localhost

Starting Nmap 5.51 ( http://nmap.org ) at 2017-01-12 15:04 CET
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000010s latency).
Other addresses for localhost (not scanned): 127.0.0.1
Not shown: 994 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
25/tcp    open  smtp
80/tcp    open  http
111/tcp   open  rpcbind
631/tcp   open  ipp
8009/tcp  open  ajp13

Nmap done: 1 IP address (1 host up) scanned in 0.04 seconds
```

Aplikacja WEB do zarządzania serwerem – Tomcat Manager

Dostęp do aplikacji Tomcat Manager

Jest to aplikacja umożliwiająca instalację, usuwanie, aktualizowanie aplikacji wdrożonych na serwer, konfigurację użytkowników, grup, ról, źródeł danych i wielu innych. Domyślnie dostęp do niej jest zablokowany. Po dodaniu odpowiedniego użytkownika dostępna będzie pod adresem:

`http://host:port/manager/html`

w moim przypadku już ze zdalnego hosta (port został przestawiony na 80, dlatego nie trzeba go już podawać):

<http://192.168.0.101/manager/html>

Jeśli teraz spróbujesz uzyskać do niej dostęp ze zdalnego hosta, powinieneś zobaczyć taki komunikat:

403 Access Denied

You are not authorized to view this page.

By default the Manager is only accessible from a browser running on the same machine as Tomcat. If you wish to modify the configuration, you must first configure the Manager application to allow access from your browser. If you have already configured the Manager application to allow access and you have used your browser's back button, returning to the [main Manager page](#). Once you return to this page, you will be able to continue using the Manager application.

If you have not changed any configuration files, please examine the file `conf/tomcat-users.xml` in your installation. For example, to add the `manager-gui` role to a user named `tomcat` with a password of `s3cret`, add the following to `conf/tomcat-users.xml`:

```
<role rolename="manager-gui"/>
<user username="tomcat" password="s3cret" roles="manager-gui"/>
```

Note that for Tomcat 7 onwards, the roles required to use the manager application were changed from the single `manager` role to the following:

- `manager-gui` - allows access to the HTML GUI and the status pages
- `manager-script` - allows access to the text interface and the status pages
- `manager-jmx` - allows access to the JMX proxy and the status pages
- `manager-status` - allows access to the status pages only

The HTML interface is protected against CSRF but the text and JMX interfaces are not. To maintain the CSRF protection:

- Users with the `manager-gui` role should not be granted either the `manager-script` or `manager-jmx` roles.
- If the text or jmx interfaces are accessed through a browser (e.g. for testing since these interfaces are intended for use by scripts), you must use the `manager-script` or `manager-jmx` roles.

For more information - please see the [Manager App HOW-TO](#).

Konfiguracja uprawnień dostępu do panelu zarządzania serwerem Tomcat

Przy próbie podłączenia się z tego samego hosta na którym zainstalowany jest Tomcat poprosi Cię o podanie użytkownika i hasła. Domyślnie Tomcat nie ma skonfigurowanych żadnych użytkowników i dlatego teraz musimy się tym zająć. Przechodzimy do edycji pliku `conf/tomcat-users.xml` i dodajemy do niego taki wpis:

```
<role rolename="manager-gui"/>
<user username="mapet" password="admin1" roles="manager-gui"/>
```

Oczywiście w miejscu hasła podajemy jakieś swoje wymyślane :) Następnie restartujemy Tomcata i ponawiamy próbę uzyskania dostępu do panelu zarządzania z lokalnego hosta. Powinniśmy zobaczyć taki mniej więcej widok:

Message: OK

Manager

List Applications HTML Manager Help Manager Help Serve

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 min
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 min
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 min

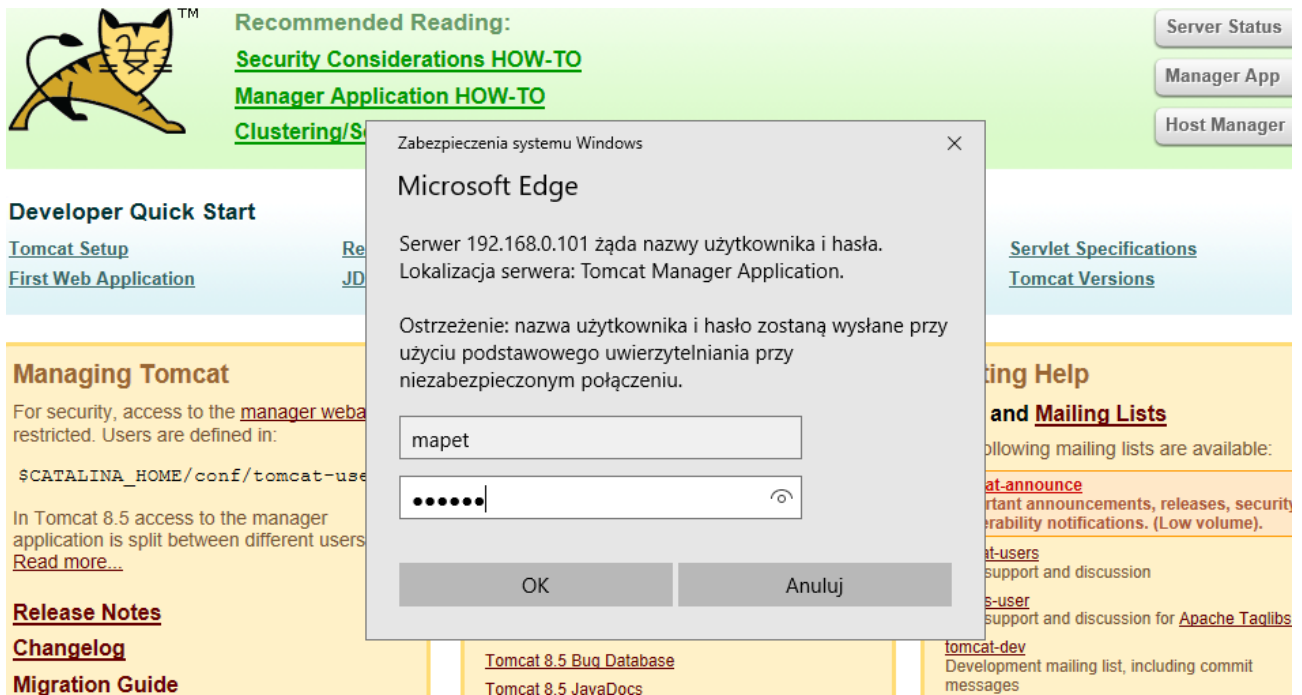
Dostęp z hosta zdalnego nadal jest zablokowany, dlatego musimy utworzyć specjalny plik konfiguracyjny który umożliwi nam dostęp do zarządzania tomcatem z użyciem Tomcat Managera przez sieć:

nano conf/Catalina/localhost/manager.xml

Wprowadzamy do niego następującą treść:

```
<Context privileged="true" antiResourceLocking="false"
  docBase="{catalina.home}/webapps/manager">
  <Valve className="org.apache.catalina.valves.RemoteAddrValve" allow="^.*$" />
</Context>
```


Nie ma już potrzeby ponownego restartowania. Przy ponownej próbie dostępu zostaniemy poproszeni o użytkownika i hasło które ustawiliśmy w tomcat-users.xml



The screenshot shows the Tomcat Manager web interface with a Windows security warning dialog box overlaid. The dialog box, titled "Zabezpieczenia systemu Windows" (Windows Security), is from Microsoft Edge and displays the following text:

Microsoft Edge

Serwer 192.168.0.101 żąda nazwy użytkownika i hasła.
Lokalizacja serwera: Tomcat Manager Application.

Ostrzeżenie: nazwa użytkownika i hasło zostaną wysłane przy użyciu podstawowego uwierzytelniania przy niezabezpieczonym połączeniu.

The dialog box contains two input fields: the first contains the username "mapet" and the second contains a masked password "•••••". At the bottom of the dialog are "OK" and "Anuluj" (Cancel) buttons.

The background shows the Tomcat Manager interface with a green header containing the Tomcat logo and navigation buttons for "Server Status", "Manager App", and "Host Manager". The main content area is divided into several sections:

- Recommended Reading:** Links to "Security Considerations HOW-TO", "Manager Application HOW-TO", and "Clustering/S".
- Developer Quick Start:** Links to "Tomcat Setup" and "First Web Application".
- Managing Tomcat:** Text explaining security restrictions and user definitions, with a link to "Read more...".
- Release Notes** and **Changelog** links.
- Migration Guide** link.
- Help and Mailing Lists:** Lists available mailing lists such as "tomcat-announce", "tomcat-users", and "tomcat-dev".

Sprawdzanie statusu serwera


Aby sprawdzić stan serwera, aktualnie podpięte sesje, środowisko w którym działa Tomcat oraz informacje o samym Tomcacie, wybieramy przycisk „Server status” znajdujący się w prawym górnym rogu strony startowej Tomcata:

Home Documentation Configuration Examples Wiki Mailing Lists Find Help

Apache Tomcat/8.5.9

The Apache Software Foundation <http://www.apache.org/>

If you're seeing this, you've successfully installed Tomcat. Congratulations!

 **Recommended Reading:**
[Security Considerations HOW-TO](#)
[Manager Application HOW-TO](#)
[Clustering/Session Replication HOW-TO](#)

Server Status (highlighted)
Manager App
Host Manager

Developer Quick Start

Powinniśmy zobaczyć taki mniej więcej obraz:

Server Status

Manager							
List Applications	HTML Manager Help	Manager Help	Complete Server Status				

Server Information							
Tomcat Version	JVM Version	JVM Vendor	OS Name	OS Version	OS Architecture	Hostname	IP Address
Apache Tomcat/8.5.9	1.8.0_102-b14	Oracle Corporation	Linux	2.6.32-642.11.1.el6.x86_64	amd64	localhost.localdomain	127.0.0.1

JVM

Free memory: 18.00 MB Total memory: 36.42 MB Max memory: 454.37 MB

Memory Pool	Type	Initial	Total	Maximum	Used
Eden Space	Heap memory	8.00 MB	10.06 MB	125.37 MB	0.36 MB (0%)
Survivor Space	Heap memory	1.00 MB	1.25 MB	15.62 MB	0.44 MB (2%)
Tenured Gen	Heap memory	20.00 MB	25.11 MB	313.37 MB	17.63 MB (5%)
Code Cache	Non-heap memory	2.43 MB	7.62 MB	240.00 MB	7.53 MB (3%)
Compressed Class Space	Non-heap memory	0.00 MB	2.75 MB	1024.00 MB	2.56 MB (0%)
Metaspace	Non-heap memory	0.00 MB	25.25 MB	-0.00 MB	24.54 MB

"ajp-nio-8009"

Max threads: 200 Current thread count: 0 Current thread busy: 0 Keep alive sockets count: 0
Max processing time: 0 ms Processing time: 0.0 s Request count: 0 Error count: 0 Bytes received: 0.00 MB Bytes sent: 0.00 MB

Stage	Time	B Sent	B Recv	Client (Forwarded)	Client (Actual)	VHost	Request
P							

P: Parse and prepare request S: Service F: Finishing R: Ready K: Keepalive

"http-nio-80"

Max threads: 200 Current thread count: 10 Current thread busy: 1 Keep alive sockets count: 1
Max processing time: 895 ms Processing time: 1.68 s Request count: 44 Error count: 9 Bytes received: 0.00 MB Bytes sent: 0.33 MB

Stage	Time	B Sent	B Recv	Client (Forwarded)	Client (Actual)	VHost	Request
R	?	?	?	?	?	?	
R	?	?	?	?	?	?	
S	12 ms	0 KB	0 KB	192.168.0.100	192.168.0.100	192.168.0.101	GET /manager/status HTTP/1.1
R	?	?	?	?	?	?	
R	?	?	?	?	?	?	

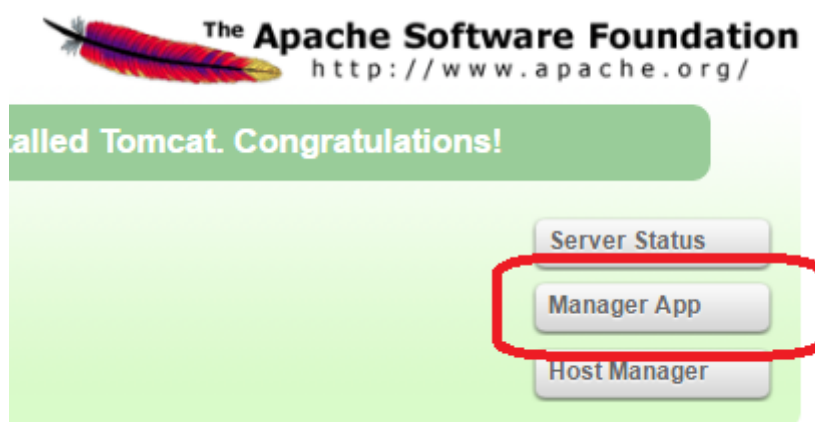
P: Parse and prepare request S: Service F: Finishing R: Ready K: Keepalive

Copyright © 1999-2016, Apache Software Foundation

W górnej sekcji zobaczymy informacje o wersji Tomcata, środowiska javy, typie i wersji systemu operacyjnego. Mniej więcej pośrodku są informacje na temat środowiska Javy w ramach którego działa Tomcat. Tutaj znajdziesz informacje na temat przydzielonej do niego ilości pamięci. Na samym końcu znajdziesz listę podpiętych sesji.

Panel zarządzania aplikacjami

Poniżej przycisku „Server Status” na stronie startowej serwera Tomcat znajdziesz przycisk „Manager App”.



Znajdziesz tutaj listę zainstalowanych aplikacji :

Applications					
Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

Możesz więc zobaczyć co jest już wdrożone na serwerze, zatrzymać, przeładować lub odinstalować aplikację.

Poniżej znajduje się także sekcja umożliwiająca wdrożenie nowej aplikacji:

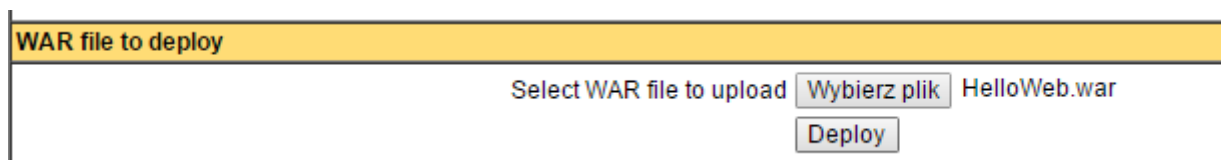
Deploy	
Deploy directory or WAR file located on server	
Context Path (required):	<input type="text"/>
XML Configuration file URL:	<input type="text"/>
WAR or Directory URL:	<input type="text"/>
	<input type="button" value="Deploy"/>
WAR file to deploy	
Select WAR file to upload	<input type="button" value="Wybierz plik"/> Nie wybrano pliku
	<input type="button" value="Deploy"/>

Wdrażanie aplikacji na serwer

Wdrażanie pliku WAR z użyciem Tomcat Managera

Aby wdrożyć plik WAR z użyciem Tomcat Managera przechodzimy do managera aplikacji (wybieramy Manager App ze strony głównej Tomcata), przechodzimy do sekcji „Deploy”.

Następnie przyciskamy „wybierz plik” i wybieramy archiwum war które chcemy zainstalować. Następnie naciskamy przycisk „Deploy”. Aplikacja powinna automatycznie zostać uruchomiona.



Możemy teraz kliknąć na link do aplikacji znajdujący się na liście aplikacji by do niej przejść:

/HelloWeb	None specified	true	0	Start Stop Reload Undeploy
				Expire sessions with idle ≥ 30 minutes
				Start Stop Reload Undeploy

Na potrzeby tego przykładu stworzyłem bardzo prostą aplikację i wdrożyłem ją w wyżej opisany sposób. Aplikacja znajdzie się pod adresem stanowiącym nazwę hosta z Tomcatem, ewentualnie port, znaku / i nazwy tej aplikacji. Wielkość liter ma znaczenie!



no siema...

Aplikacja powinna zostać po wdrożeniu automatycznie uruchomiona, jeśli jednak na liście aplikacji przy tej apce dostępny jest przycisk „Start”, to oznacza to że nie została ona uruchomiona i należy zajrzeć do logów w celu znalezienia przyczyny takiego stanu rzeczy.

Nasz plik wylądował w podkatalogu webapps Tomcata, a także został w tym samym katalogu rozpakowany.

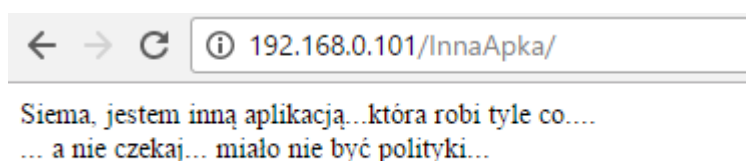
```
[root@localhost apache-tomcat-8.5.9]# ls webapps/  
docs  examples  HelloWeb  HelloWeb.war  host-manager  manager  ROOT  
[root@localhost apache-tomcat-8.5.9]#
```

Automatyczne rozpakowywanie archiwum WAR

To że aplikacja jest rozpakowywana z pliku WAR jest zasługą domyślnego ustawienia parametru `unpackWARs` w pliku `conf/server.xml`

```
<Host name="localhost"  appBase="webapps"  
      unpackWARs="true"  autoDeploy="true">
```

Jeśli ta wartość jest ustawiona na `true`, będzie działało się tak jak przed momentem. Jeśli zmienisz to ustawienie na `false`, plik war nie będzie rozpakowywany a żądane pliki będą czytane bezpośrednio z archiwum. Na potrzeby pokazania tej funkcjonalności zmieniłem parametr `unpackWARs` na `false`, stworzyłem kolejną aplikację i ją wdrożyłem:

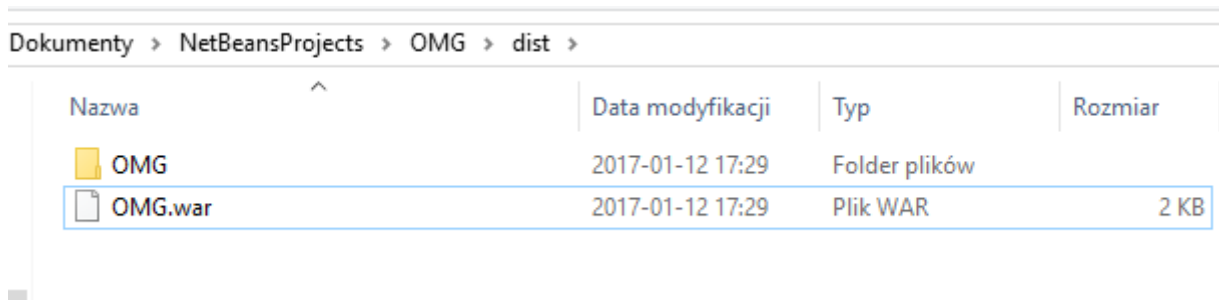


Tym razem archiwum nie zostało jednak rozpakowane:

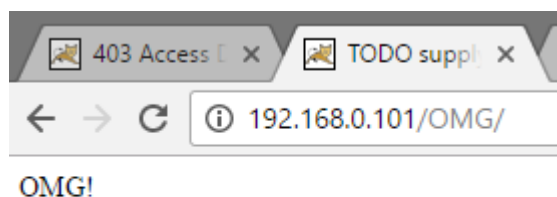
```
[root@localhost apache-tomcat-8.5.9]# ls webapps/  
docs  examples  HelloWeb  HelloWeb.war  host-manager  InnaApka.war  manager  ROOT  
[root@localhost apache-tomcat-8.5.9]#
```

Wdrażanie bez użycia aplikacji Tomcat Manager

Aplikacje możemy wdrażać również ręcznie, bez korzystania z aplikacji Tomcat Manager. Wystarczy umieścić aplikację w podkatalogu WEBAPPS serwera Tomcat, wszystko jedno czy prześlesz w postaci spakowanego archiwum WAR czy rozpakujesz WARa do katalogu i wtedy prześlesz.



Aplikacja powinna zostać wdrożona od razu, bez potrzeby restartowania serwera.



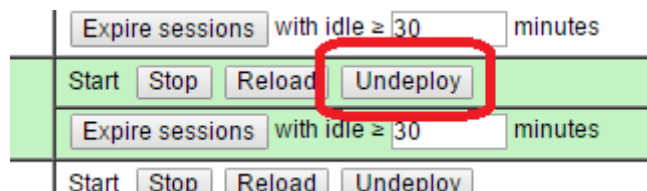
Wdrożone w ten sposób aplikacje również będą widoczne na liście w managerze aplikacji:

Manager					
List Applications	HTML Manager Help		Manager Help		S
Applications					
Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/HelloWeb	None specified		true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/InnaApka	None specified		true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/OMG	None specified		true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

Ciekawostka – możesz wyedytować pliki w aplikacji „na żywo” na serwerze, a wszelkie zmiany zostaną od razu zaimplementowane.

Deinstalacja aplikacji

Aplikację możesz odinstalować wybierając przycisk „Undeploy”. W managerze aplikacji.



Niezależnie od sposobu wdrożenia kasowany jest także fizycznie katalog aplikacji i plik WAR z katalogu WEBAPPS. Wynika to z faktu automatycznego wdrażania aplikacji które znajdują się w tym katalogu. Gdyby tak nie było, po chwili nasza aplikacji znowu by się sama wdrożyła ze względu na ustawienie parametr autoDeploy w pliku server.xml

```
[root@localhost webapps]# ls
docs  examples  host-manager  manager  OMG.war  ROOT
[root@localhost webapps]#
```

```
<Host name="localhost"  appBase="webapps"
      unpackWARs="true"  autoDeploy="true">
```


Możesz też skasować fizycznie katalog aplikacji i plik WAR z katalogu WEBAPPS. W takim przypadku link do aplikacji również samoczynnie znika z listy dostępnych aplikacji.

Applications		
Path	Version	Display Name
/	<i>None specified</i>	Welcome to Tomcat
<u>/docs</u>	<i>None specified</i>	Tomcat Documentation
<u>/examples</u>	<i>None specified</i>	Servlet and JSP Examples
<u>/host-manager</u>	<i>None specified</i>	Tomcat Host Manager Application
<u>/manager</u>	<i>None specified</i>	Tomcat Manager Application

Deploy

Zmiana adresu aplikacji

Zauważ że aplikacja domyślnie znajduje się pod adresem zgodnym z jej nazwą. Jeśli chciałbyś by Twoja aplikacja była dostępna pod innym adresem, wprowadź analogiczny do poniższego wpis do pliku server.xml:

```
<Context docBase="OMG" path="/innyadres" />
```

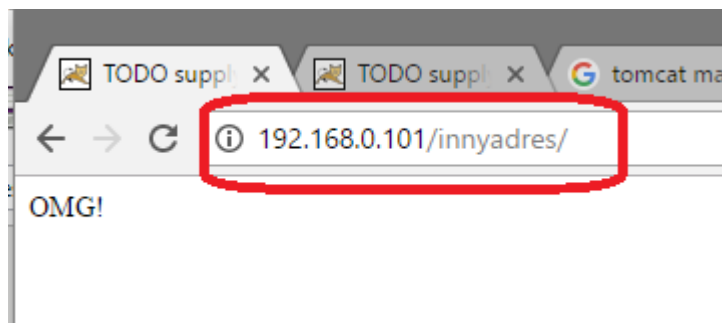
w obrębie sekcji „HOST” tak by w efekcie uzyskać zapis jak niżej:

```
<Host name="localhost" appBase="webapps"
      unpackWARs="false" autoDeploy="true">
  <Context docBase="OMG" path="/innyadres" />
</Host>
```

Po tej zmianie koniecznie trzeba zrestartować serwer by mógł wczytać nowe ustawienia. Aplikacja będzie dostępna zarówno pod dotychczasowym adresem „OMG” jak i nowym wskazanym „innyadres”:

Applications		
Path	Version	Display Name
/	None specified	Welcome to Tomcat
<u>/OMG</u>	None specified	
<u>/docs</u>	None specified	Tomcat Documentation
<u>/examples</u>	None specified	Servlet and JSP Examples
<u>/host-manager</u>	None specified	Tomcat Host Manager Application
<u>/innyadres</u>	None specified	
<u>/manager</u>	None specified	Tomcat Manager Application

Sprawdźmy jeszcze jak się ładuje:



Wdrażanie jako domyślna główna aplikacja

Ustawienie wybranej aplikacji jako głównej dla Tomcata sprowadza się do podobnej zmiany co wdrożenie aplikacji pod innym adresem. Edytujemy plik `conf/server.xml` i w elemencie `Context` (jeśli go nie ma to go dodajemy) wprowadzamy pusty ciąg w miejsce „`path`” jak widać poniżej:

```
unpackWARs="false" autoDe
<Context docBase="OMG" path="" />
<!-- SingleSignOn valve, shar
```

Od tej pory po wprowadzeniu adresu serwera i portu (a jeśli Tomcat działa na porcie 80 to także bez portu) powinniśmy zobaczyć aplikację znajdującą się w podkatalogu `OMG` w `webapps`:



Alternatywnie możesz rozpakować aplikację do podkatalogu `ROOT` wewnątrz katalogu `webapps`. Efekt będzie ten sam. Pamiętaj o restarcie Tomcata.

Zmiana adresu aplikacji wdrożonej jako archiwum WAR bez rozpakowywania

Tak jak zmienialiśmy adres dostępowy do rozpakowanej aplikacji, tak możemy to zrobić dla aplikacji wdrożonej jako WAR bez jej rozpakowywania. Zmiana polega na wprowadzeniu do conf/server.xml takiego samego wpisu jak wcześniej, z tą jednak różnicą że tym razem jako docBase podajemy plik WAR a nie nazwę katalogu. Na poniższej ilustracji zaznaczyłem też fragment odnoszący się do rozpakowywania plików WAR w ogóle przy okazji wdrożenia. Najlepiej przy takiej konfiguracji ustawić rozpakowywanie plików WAR na false, aby nie pomylić później faktycznego źródła wszelkich plików aplikacji.

```
<Host name="localhost" appBase="webapps"
      unpackWARs="false" autoDeploy="true">
<Context docBase="OMG" path="" />
<Context docBase="HelloWeb.war" path="/hellomoto"/>
```

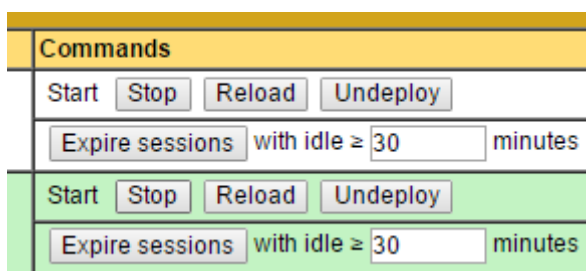
Zatrzymywanie i startowanie aplikacji z użyciem Tomcat Manager'a.

Serwer Tomcat umożliwia zatrzymywanie i uruchamianie wybranych aplikacji, bez potrzeby zatrzymywania i uruchamiania całego serwera. Na potrzeby przykładu stworzyłem przykładową aplikację i wdrożyłem ją na serwer:

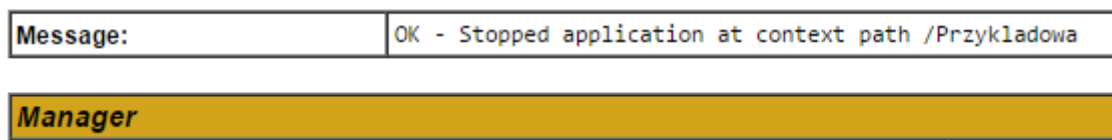


No cześć, to ja. Przykładowa aplikacja. Mapet mnie napisał.

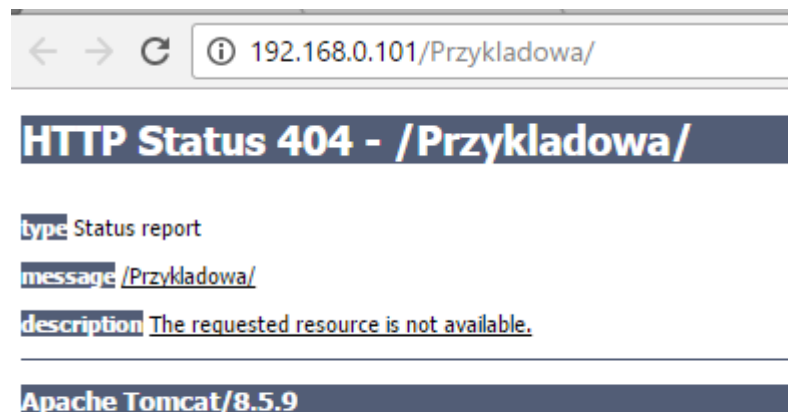
Na liście aplikacji po prawej stronie znajduje się kolumna „Commands”. Klikamy „Stop” przy wybranej aplikacji.



W górnej części panelu powinniśmy zobaczyć taki komunikat:



Po zatrzymaniu aplikacji i próbie dostępu do niej, otrzymujemy komunikat taki jakby nie było aplikacji w ogóle.



Naciśnięcie przycisku „START” obok użytego przed momentem przycisku „STOP” spowoduje ponowne uruchomienie aplikacji. Jeśli aplikacja zostaje zatrzymana, to po ponownym uruchomieniu serwera taka NIE pozostanie! Zostanie uruchomiona razem z serwerem!

Sesje aplikacji i ich rozłączanie

Z poziomu Tomcat Managera możemy także podejrzeć aktualnie podpięte do aplikacji sesje. Na liście aplikacji mamy kolumnę „Sessions”.

Applications				
Path	Version	Display Name	Running	Sessions
/	None specified	Welcome to Tomcat	true	0
/Zepsuta	None specified		true	0
/docs	None specified	Tomcat Documentation	true	0
/examples	None specified	Servlet and JSP Examples	true	0
/host-manager	None specified	Tomcat Host Manager Application	true	0
/manager	None specified	Tomcat Manager Application	true	1

Deploy

Kiedy klikniemy na liczbę przy wybranej aplikacji, przejdziemy do listy aktualnie trwających sesji.

Sessions Administration for /manager

Tips:

- Click on a column to sort.
- To view a session details and/or remove a session attributes, click on its id.

Active HttpSession informations

Session Id	Type	Guessed Locale	Guessed User name	Creation Time	Last Accessed Time	Used Time	Inactive Time	TTL
<input type="checkbox"/> E08655D9343B6BE191330DD90656B96B	Primary		mapet	2017-01-24 13:17:37	2017-01-24 13:17:53	00:03:27	00:00:00	00:29:59

Invalidate selected Sessions

Return to main page

Z poziomu tego widoku możemy także przerwać wybrane sesje. Wystarczy zaznaczyć checkbox obok wybranej sesji, a następnie użyć przycisku „Invalidate selected sessions”

Active HttpSession informations

Session Id	Type	Guessed Locale	Guessed User name
<input checked="" type="checkbox"/> E08655D9343B6BE191330DD90656B96B	Primary		mapet

Invalidate selected Sessions

Return to main page

Każda sesja ma swój określony czas „ważności”. Domyślnie jest to 30 minut, tę własność konfiguruje się w pliku web.xml wewnątrz aplikacji. Po upływie tego czasu sesje się „unieważniają” co sprowadzać się będzie do np. wylogowania z aplikacji i konieczności ponownego zalogowania. Z poziomu panelu managera Tomcata możemy „unieważnić” wszystkie sesje które pozostają bezczynne od wskazanego czasu. Wystarczy w polu „idle time” wpisać liczbę minut, a następnie kliknąć „Expire sessions”. Wszystkie sesje bezczynne od przynajmniej wskazanego czasu zostaną „unieważnione”.

Sessions	Commands
0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

Konsola tekstowa i wykorzystanie skryptów do zarządzania serwerem

Uprawnienia

Istnieje alternatywna tekstowa wersja Tomcat Manager'a służąca do zarządzania aplikacjami z użyciem wywołań po http. Idealnie nadaje się do podpięcia zarządzania aplikacjami poprzez skrypty. Aby ta funkcjonalność działała, musimy użytkownikowi z którego korzystamy dodać rolę manager-script. Robimy to edytując skrypt tomcat-users.xml znajdujący się w podkatalogu conf.

W naszym przypadku będzie się to sprowadzać do dodania drobnego wpisu do wcześniej utworzonego użytkownika:

```
<role rolename="manager-gui"/>
<user username="mapet" password="admin1" roles="manager-gui,manager-script"/>
<!--
```

Po tej zmianie należy oczywiście zrestartować serwer Tomcata żeby zmiany zostały uwzględnione.

Zatrzymywanie i uruchamianie aplikacji z poziomu konsoli / skryptu

Zatrzymywanie i uruchamianie aplikacji sprowadza się do wywołania odpowiednio skonstruowanego adresu URL. Można to wykonać choćby narzędziem wget (dostępnym w systemach Linux). Konstrukcja wywołania:

```
wget „http://uzytkownik:haslo@adres\_ip:port/manager/text/<komendy>
```

Aby zatrzymać aplikację wywołujemy :

```
wget „http://uzytkownik:haslo@adres\_ip:port/manager/text/stop?path=Aplikacja” -O -
```

Aby uruchomić zatrzymaną:

```
wget „http://uzytkownik:haslo@adres\_ip:port/manager/text/start?path=Aplikacja” -O -
```

```
wget "http://mapet:admin1@192.168.0.101:80/manager/text/stop?path=/Przykladowa" -O -
```

```
[root@localhost apache-tomcat-8.5.9]# wget "http://mapet:admin1@192.168.0.101:80/manager/text/stop?path=/Przykladowa" -O -
--2017-01-24 14:19:48-- http://mapet:*password*@192.168.0.101/manager/text/stop?path=/Przykladowa
Connecting to 192.168.0.101:80... connected.
HTTP request sent, awaiting response... 401
Reusing existing connection to 192.168.0.101:80.
HTTP request sent, awaiting response... 200
Length: unspecified [text/plain]
Saving to: "STDOUT"

[<=>
K - Stopped application at context path /Przykladowa
[ <=>

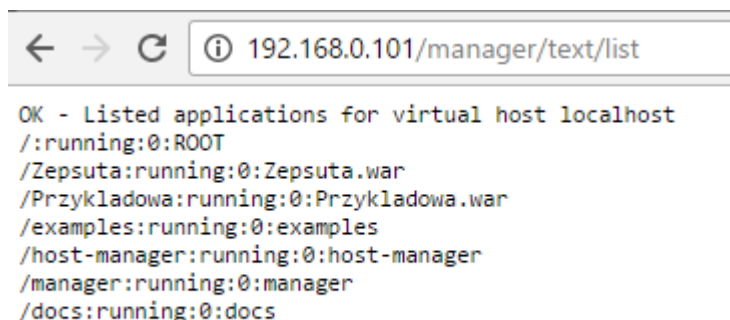
2017-01-24 14:19:48 (12.8 MB/s) - written to stdout [54]
```

Lista wdrożonych aplikacji wraz z informacją o ilości sesji i stanie uruchomienia z poziomu konsoli

W trybie tekstowym możemy uzyskać listę wszystkich aplikacji i ich stanu korzystając z adresu wg formatu:

http://adres_ip:port/manager/text/list

Taki wynik można łatwo przeparsować choćby narzędziem GREP:



```
OK - Listed applications for virtual host localhost
/:running:0:ROOT
/Zepsuta:running:0:Zepsuta.war
/Przykladowa:running:0:Przykladowa.war
/examples:running:0:examples
/host-manager:running:0:host-manager
/manager:running:0:manager
/docs:running:0:docs
```

Przeładowanie aplikacji z poziomu konsoli

Przeładowanie aplikacji, podobnie jak jej zatrzymanie lub uruchomienie sprowadza się do wywołania odpowiednio skonstruowanego adresu URL. Można to wykonać choćby narzędziem wget (dostępnym w systemach Linux). Konstrukcja wywołania:

wget „http://uzytkownik:[haslo@adres_ip](#):port/manager/text/reload?path=Aplikacja” -O -

wget "http://mapet:admin1@192.168.0.101:80/manager/text/reload?path=/Przykladowa" -O -

```
[root@localhost apache-tomcat-8.5.9]# wget "http://mapet:admin1@192.168.0.101:80/manager/text/reload?path=/Przykladowa" -O -
--2017-01-24 16:16:00-- http://mapet:*password*@192.168.0.101/manager/text/reload?path=/Przykladowa
Connecting to 192.168.0.101:80... connected.
HTTP request sent, awaiting response... 401
Reusing existing connection to 192.168.0.101:80.
HTTP request sent, awaiting response... 200
Length: unspecified [text/plain]
Saving to: "STDOUT"

[<=>
K - Reloaded application at context path /Przykladowa
[ <=>

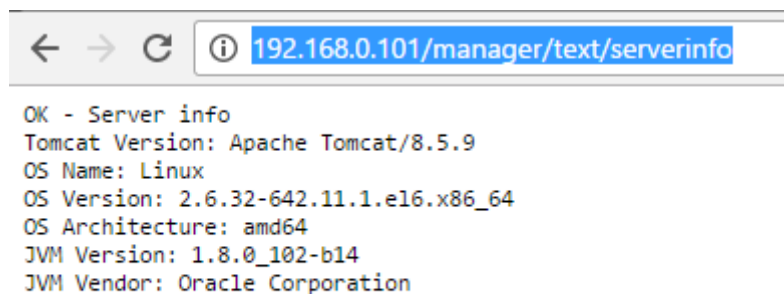
2017-01-24 16:16:00 (13.8 MB/s) - written to stdout [55]
```

Sprawdzanie statusu serwera w trybie tekstowym

Aby uzyskać informacje o stanie serwera wystarczy wywołać adres wg zasady:

http://adres_ip:port/manager/text/serverinfo

Dane zwrotne dostajemy w postaci tekstowej, tak więc podobnie jak w przypadku listy aplikacji w trybie tekstowym, takie dane łatwo będzie dalej obrobić.

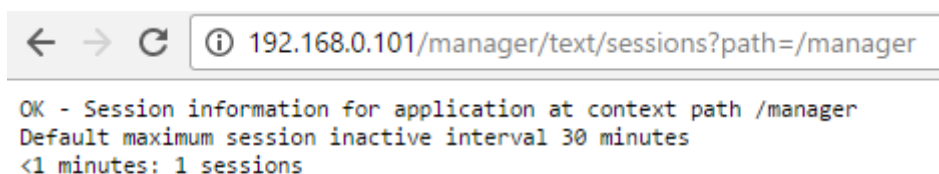


```
← → ↻ ⓘ 192.168.0.101/manager/text/serverinfo
OK - Server info
Tomcat Version: Apache Tomcat/8.5.9
OS Name: Linux
OS Version: 2.6.32-642.11.1.el6.x86_64
OS Architecture: amd64
JVM Version: 1.8.0_102-b14
JVM Vendor: Oracle Corporation
```

Sprawdzanie w trybie tekstowym sesji wskazanej aplikacji

Aby uzyskać informacje o stanie serwera wystarczy wywołać adres wg zasady:

http://adres_ip:port/manager/text/sessions?path=/nazwaaplikacji

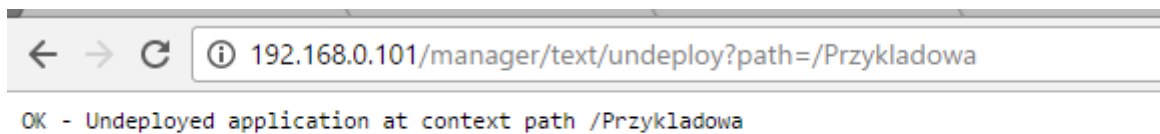


```
← → ↻ ⓘ 192.168.0.101/manager/text/sessions?path=/manager
OK - Session information for application at context path /manager
Default maximum session inactive interval 30 minutes
<1 minutes: 1 sessions
```

Odinstalowywanie aplikacji z poziomu konsoli

Aby odinstalować aplikację należy wywołać URL wg zasady:

`http://adres_ip:port/manager/text/undeploy?path=/nazwaaplikacji`



Składowe serwera

Katalogi serwera

bin	Wszelkie pliki binarne, w tym pliki odpowiedzialne za uruchamianie i zatrzymywanie serwera.
conf	Pliki konfiguracji serwera
lib	Miejsce na zewnętrzne dołączane do serwera biblioteki
logs	Miejsce składowania logów
webapps	Miejsce w którym przechowywane są zainstalowane aplikacje
work	Skompilowane wersje plików JSP

Najważniejsze pliki i ich zawartość

Server.xml	Główny plik konfiguracji serwera
Tomcat-users.xml	Role, użytkownicy i hasła
Catalina.policy	Zasady bezpieczeństwa
Context.xml	Ustawienia kontekstów. Jeśli ustawisz tutaj jakieś parametry będą one dotyczyły wszystkich aplikacji na serwerze. Przykładowo możesz tutaj skonfigurować parametr reloadable – będzie on dotyczył wszystkich wdrożonych aplikacji.
Catalina.out	Główny log serwera

Pliki konfiguracyjne aplikacji

Poza wspólnymi ustawieniami dla wszystkich aplikacji, konfigurowanymi w pliku context.xml, każda aplikacja może mieć także swój osobny plik konfiguracyjny. Położenie tego pliku będzie zależne od parametrów ENGINE, HOST i CONTEXT określanych w pliku server.xml.

Przykładowo wdrożona aplikacja Manager posiada taki plik, a jeśli nie zmienialiśmy domyślnej konfiguracji w.w elementów to jej plik konfiguracyjny znajdzie się w lokalizacji :

katalog_tomcata/conf/Catalina/localhost/manager.xml

```
[root@localhost apache-tomcat-8.5.9]# ls conf/Catalina/localhost
manager.xml
[root@localhost apache-tomcat-8.5.9]# nano conf/Catalina/localhost/manager.xml
[root@localhost apache-tomcat-8.5.9]# █
```

Zawartość przykładowego pliku:

```
root@localhost:/home/andrew/soft/apache-tomcat-8.5.9
GNU nano 2.0.9 File: cor
Context privileged="true" antiResourceLocking="false"
  docBase="${catalina.home}/webapps/manager">
  <Valve className="org.apache.catalina.valves.RemoteAddrValve" allow="^.*$" />
</Context>
```

Konfiguracja serwera Tomcat

SERVER.XML

To właśnie w tym pliku znajduje się większość istotnej konfiguracji. Sam plik jest podzielony na pewne sekcje, ogólny zarys zagnieżdżeń sekcji prezentuje się tak:

```
<Server>
  <Service>
    <Connector/>
    <Connector/>
    <Engine>
      <Host>
        <Context>
        </Context>
        <Context>
        </Context>
      </Host>
    </Engine>
  </Service>
  <Service>
  </Service>
</Server>
```


Element Server

Nadrzędny element struktury. Musi on być główną strukturą, może występować tylko jeden. Reprezentuje cały serwer. Może zawierać jedną lub więcej usług (<Service>), musi posiadać przynajmniej jedną.

Element Service

Zawiera jeden lub więcej elementów <Connector> i tylko jeden element <Engine>

Element Connector

Reprezentuje klasę odpowiedzialną za odpowiadanie na żądania. Kilka ciekawych parametrów:

Nazwa parametru	Sposób wprowadzenia	Opis
port	Port="80"	Port nasłuchu connectora
maxPostSize	maxPostSize="10m"	Maksymalna wielkość przesyłanego przez POST formularza. Jeśli ustawisz 0, nie będzie żadnego limitu. Domyślny limit to 2MB.
uriEncoding	uriEncoding="utf-8"	Określa kodowanie znaków w adresach URL. Domyślnie ustawione na ISO-8859-1
address	Address="1.2.3.4"	Określa adres na którym connector ma nasłuchiwać. Użyteczne gdy serwer ma kilka kart sieciowych, a usługi Tomcata mają być dostępne tylko przez jedną z nich. Domyślnie nie jest ustaiwony.
compression	compression="off"	Umożliwia włączenie kompresji dla przesyłanych przez dany Connector danych. Służy do ograniczania zużycia transferu. Możliwe są trzy ustawienia. OFF – wyłączone, ON – włączone dla treści tekstowych, FORCE – kompresja dla wszystkich

		elementów, nie tylko tekstowych. Domyślnie wyłączony
connectionTimeout	connectionTimeout="60000"	Maksymalny czas dla wywołania. Jeśli nie zostanie ono obsłużone przez wskazany czas, jest przerywane. Wartość wyrażona w milisekundach. Domyślnie 60s. Ustawienie wartości -1 powoduje całkowite wyłączenie limitów.
maxThreads	MaxThreads="200"	Maksymalna ilość wątków które mogą być utworzone do obsługi żądań przychodzących na dany Connector. Domyślnie 200.
acceptCount	AcceptCount="10"	Maksymalna ilość żądań która może zostać skolejkowana na danych Connectorze. Domyślnie 10.

Element Engine

Obsługuje wszystkie żądania przychodzące do elementów <Connector> należące do nadrzędnej usługi (elementu <Service>).

Element Host

Reprezentuje wirtualny host wewnątrz serwera. Każdy host może być nadrzędnym elementem dla jednego lub więcej elementów <Context> reprezentujących aplikacje. Ma on kilka interesujących parametrów – większość z nich została omówiona już wcześniej.

Nazwa parametru	Sposób wprowadzenia	Opis
Name	name="localhost"	<p>Nazwa wirtualnego hosta. Jeśli masz przekierowane kilka domen na ten sam adres IP serwera, możesz utworzyć kilka elementów HOST o różnych parametrach name. Dzięki temu w zależności od adresu wywołania, żądanie będzie przekazywane do odpowiedniego wirtualnego hosta. Co nam to daje? Obsługę wielu domen na jednym Tomcacie, a także możliwość stosowania subdomen na zasadzie: blog.jsystems.pl forum.jsystems.pl etc.</p> <p>Gdybyśmy coś takiego chcieli osiągnąć, nasza konfiguracja powinna wyglądać mniej więcej tak:</p> <pre><Host name="blog.jsystems.pl" appBase="webapps"> <Context path="" docBase="/blog/" /> </Host></pre> <pre><Host name="forum.jsystems.pl" appBase="webapps"> <Context path="" docBase="/forum/" /> </Host></pre> <p>Ważne jest by wartość parametru name pokrywała się z wartością parametru defaultHost dla podrzędnego elementu Engine. Jeśli takowy nie jest ustawiony, domyślnie przyjmuje localhost. Wartość localhost zarówno w elemencie host jak i engine obsługuje wszystkie żądania.</p>
appBase	appBase="webapps"	Deklaruje nazwę katalogu w którym

		serwer Tomcat ma poszukiwać plików aplikacji wdrożonych w ramach tego wirtualnego hosta. Domyślnie jest to katalog webapps. Może to być zarówno ścieżka bezwzględna jak i względna do katalogu instalacji Tomcata (CATALINA_HOME).
autoDeploy	autoDeploy="true"	Określa czy serwer Tomcat ma automatycznie wdrażać aplikacje na danym hoście wirtualnym. Domyślnie ustawiony na true.
deployOnStartup	deployOnStartup="true"	Określa czy znalezione przy starcie serwera aplikacje wewnątrz katalogu określonego przez parametr appBase mają być automatycznie wdrażane czy nie. Domyślnie ustawiony na true.
unpackWARs	unpackWARs="false"	Określa czy aplikacje wdrożone pod postacią plików WAR mają być rozpakowywane podczas wdrażania (w ramach danego wirtualnego hosta) i pliki mają być podawane z rozpakowanego katalogu, czy też mają być pobierane bezpośrednio z pliku war. Domyślnie ustawiony na true.
workDir	workDir="work"	Wskazuje katalog roboczy na użytek danego wirtualnego hosta. Domyślnie ustawiony na podkatalog work serwera Tomcat.

Element Context

Każdy element Context reprezentuje jedną aplikację uruchomioną w ramach hosta na serwerze Tomcat. Elementów Context wewnątrz elementu Host może być dowolna ilość. Zasadniczo ilekroć będę pisał o context – możesz to rozumieć przez jakąś konkretną aplikację. Niektóre z parametrów są takie same jak dla elementu Host który jest dla elementu Context nadrzędny. Jeśli zdublujemy parametry to to który obowiązuje zależy od ustawienia parametru „override” elementu Context. Ten jest domyślnie ustawiony na true. Wynika z tego, że nawet jeśli będziemy mieć ten sam parametr np. „unpackWar” ustawiony i dla Host i dla Context to obowiązuje ten ustawiony dla Context, ponieważ przesłania ten zadeklarowany w elemencie Host. Będzie tak tak długo, jak długo nie zmienimy domyślnej wartości parametru override (który nawet nie musi być wprowadzony) z true na false. Po takiej zmianie ważniejsze będzie ustawienie dla elementu Host i ono będzie obowiązywać, niezależnie od ustawień elementu Context.

Nazwa parametru	Sposób wprowadzenia	Opis
docBase	docBase=„wyginamSmialoCialo”	Wskazuje katalog (będący podkatalogiem katalogu ustawionego w parametrze appBase elementu Host) w którym serwer ma szukać plików danej aplikacji. Jeśli nie ustawimy tego parametru, serwer będzie poszukiwał katalogu o nazwie zgodnej z nazwą aplikacji.
path	Path=„/OMG”	Względny adres pod jakim ma być dostępna aplikacja. Jeśli tego nie ustawimy, aplikacja będzie dostępna pod adresem zgodnym z nazwą katalogu aplikacji. Wartość path musi być unikalna w ramach całego elementu HOST.
reloadable	reloadable=„true”	Jeśli jest ustawiony na true, Tomcat będzie cyklicznie sprawdzał zawartość podkatalogów aplikacji WEB-INF/classes i WEB-INF/lib pod kątem ewentualnych zmian. Jeśli takowe zmiany wykryje – przeładuje aplikację.
override	override=„true”	Określa czy ustawienia dla elementu Context mają przesłaniać ustawienia

		nadrzędnego elementu Host.
unpackWar	unpackWar="true"	Jak w elemencie Host, ale dotyczy tylko danej aplikacji.
workDir	workDir="iHaveNoTimeToWork"	Jak w elemencie Host, ale dotyczy tylko danej aplikacji.
cachingAllowed	cachingAllowed="true"	Od ustawienia tego elementu zależy czy statyczne zasoby aplikacji (takie jak obrazki czy pliki CSS) będą przez Tomcata buforowane.

Debugowanie serwera i rozwiązywanie problemów

Rodzaje logów serwera Tomcat

Wszystkie logi serwera Tomcat domyślnie znajdują się w podkatalogu „logs”. Znajdziemy tam pliki o strukturach nazw:

- catalina.out
- catalina.yyyy-mm-dd.txt
- localhost.yyyy-mm-dd.txt
- localhost_access.log.yyyy-mm-dd.txt
- host-manager.yyyy-mm-dd.txt
- manager.yyyy-mm-dd.txt

Plik o nazwie catalina.out

Główny log serwera Tomcat. To właśnie tutaj szukamy informacji o pojawiających się błędach serwera. Wyjątki rzucone przez same aplikacje nie znajdują się w tym pliku. Ponieważ ten plik nie jest w żaden sposób czyszczony, potrafi urosnąć do naprawdę dużych rozmiarów. W takiej sytuacji wystarczy go skasować, a następnie zrestartować serwer Tomcata. Plik zostanie utworzony na nowo, pusty.

Pliki o formacie catalina.yyyy-mm-dd.txt

Zasadniczo znajdziemy tak takie same informacje jak w pliku catalina.out. Różnica polega na tym, że w tych plikach logi są rozdzielone na poszczególne dni.

Pliki o formacie localhost.yyyy-mm-dd.txt

Tutaj znajdziemy informacje o błędach aplikacji. Wyjątki rzucone przez same aplikacje nie znajdują się w pliku catalina.out. Jeśli więc będzie jakiś kłopot z działaniem którejś aplikacji a w pliku catalina.out nie znajdziemy żadnych informacji, szukajmy ich w pliku zawierającym w nazwie datę dzisiejszą (lub tego dnia kiedy problemy się pojawiły).

Pliki o formacie localhost_access.log.yyyy-mm-dd.txt

Tutaj znajdziemy szczegółowe dane na temat dostępów do wszystkich aplikacji, podstron czy zasobów statycznych – wraz z dokładnym czasem oraz adresem IP klienta i numerami statusów zwracanymi przez Tomcata. Te pliki stają się bardzo przydatne w 2 sytuacjach. Po pierwsze na potrzeby analizy powłamaniowej. Po drugie gdy szukamy wszystkich brakujących elementów – tj. takich gdzie ktoś próbował uzyskać do nich dostęp, ale dostał w zwrocie błąd numer 401 (not found). Wystarczy przeparsować te logi pod kątem wystąpienia wartości 401.

Pliki o formacie manager.yyyy-mm-dd.txt

Tutaj znajdziemy listing wszystkich dostępów do aplikacji manager – zarówno przyjaznej HTMLowej wersji, jak i konsoli tekstowej. Można łatwo znaleźć (wraz z adresami IP) kto i co robił z użyciem tej aplikacji.

Szukanie przyczyn problemów

Jeśli jakaś aplikacja zgłasza błąd 500 (Internal Server Error) lub serwer nie działa właściwie, powinniśmy zajrzeć w pierwszej kolejności do pliku catalina.out znajdującego się w katalogu logs serwera. Jest to główny log serwera Tomcat. Znajdą się tam informacje o błędach samego serwera Tomcat. Poniżej przedstawiam przykład takiej analizy. Zbudowałem najprostszą aplikację zawierającą serwlet. Poniżej kod realizowany przez ten serwlet. Poniższa metoda obsługuje wywołanie podstrony /Zepsuj

```
18 public class Zepsuj extends HttpServlet {
19
20     @Override
21     protected void doGet(HttpServletRequest request, HttpServletResponse response)
22         throws ServletException, IOException {
23         response.setContentType("text/html;charset=UTF-8");
24         try (PrintWriter out = response.getWriter()) {
25             out.println("<!DOCTYPE html>");
26             out.println("<html>");
27             out.println("<head>");
28             out.println("<title>Servlet Zepsuj</title>");
29             out.println("</head>");
30             out.println("<body>");
31             out.println("<h1>" + (1/0) + "</h1>");
32             out.println("</body>");
33             out.println("</html>");
34         }
35     }
```

Jak nie trudno zauważyć, następuje tutaj próba dzielenia przez zero (linia 31). Aplikację kompiluję i wrażam na serwer. Wchodzę pod adres obsługiwany przez ten serwlet:

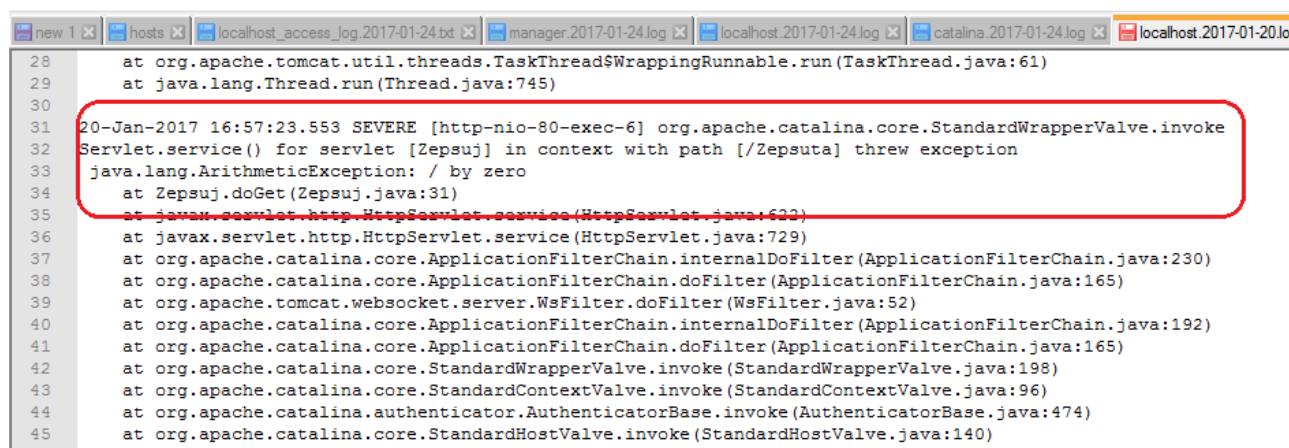


Z racji oczywistego błędu nie dostaję oczekiwanej strony internetowej. Podejrzymy logi. W systemie Linux domyślnie dostępna jest aplikacja TAIL która pozwala m.in. na podglądanie na żywo nowo pojawiających się logów. Wystarczy ją wywołać z użyciem przełącznika -f i podać ścieżkę względną lub bezwzględną pliku którego chcemy podglądać. W naszym przypadku (a jestem w głównym katalogu Tomcata, cała komenda ma kształt:

tail -f logs/catalina.out

Użytkownicy systemu Windows nie muszą się martwić, bo jest dostępna adaptacja tego programu dla ich systemu. Wystarczy wyszukać i zainstalować aplikację „Tail for Windows”.

W tym przypadku w logach catalina.out nie pojawiło się nic na temat owego błędu. Możemy więc wywnioskować, że nie jest to błąd serwera a aplikacji. Przechodzimy więc do logu o formacie „localhost.yyyy-mm-dd.log” z dzisiejszą datą i odnajdujemy taki wpis:



```
28     at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:61)
29     at java.lang.Thread.run(Thread.java:745)
30
31 20-Jan-2017 16:57:23.553 SEVERE [http-nio-80-exec-6] org.apache.catalina.core.StandardWrapperValve.invoke
32 Servlet.service() for servlet [Zepsuj] in context with path [/Zepsuta] threw exception
33 java.lang.ArithmeticException: / by zero
34     at Zepsuj.doGet(Zepsuj.java:31)
35     at javax.servlet.http.HttpServlet.service(HttpServlet.java:622)
36     at javax.servlet.http.HttpServlet.service(HttpServlet.java:729)
37     at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:230)
38     at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:165)
39     at org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)
40     at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:192)
41     at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:165)
42     at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:198)
43     at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:96)
44     at org.apache.catalina.authenticator.AuthenticatorBase.invoke(AuthenticatorBase.java:474)
45     at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:140)
```

Dla osób nie programujących w Javie – możesz po prostu wyszukać w pliku wpisu o treści „Exception” albo po prostu nazwy problematycznej aplikacji. Jak widać powyżej, z logów dowiemy się nie tylko o czasie wystąpienia błędu, ale też o rodzaju tego błędu, a nawet o linii w kodzie źródłowym która ten błąd spowodowała (zobacz w powyższym logu linię nr 34).