Maven Building a web application (cmd)

How to do it...

Start the command-line terminal and execute the following generate command:

mvn archetype:generate -DarchetypeArtifactId=maven-archetype-webapp -DartifactId=testWebApp -DgroupId=ch.hftm.webapp -Dversion=0.0-SNAPSHOT

Zuletzt geändert: Dienstag, 28. April 2015, 20:16

Create a Maven Project with CommandLine

http://websystique.com/maven/create-a-maven-project-with-commandline/

mvn archetype:generate -DgroupId=gscs.ch-DartifactId=TruckersForum -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

Einstellungen

2521 Web Applikationen I

Maven Running a web application (Jetty)

Jetty Plugin

Maven can automate the process of deploying, cleaning, and redeploying the WAR on a developer machine web application server, making the entire process less cumbersome for the developer. This is done with the Jetty plugin.

Getting ready...

Let us add the plugin element shown in the following code to your web application project's build configuration:

```
<build>
  <plugins>
    ...
    <plugin>
      <groupId>org.mortbay.jetty</groupId>
      <artifactId>maven-jetty-plugin</artifactId>
      <version>6.1.26</version>
    </plugin>
    ...
  </plugins>
</build>
```

How to do it...

Once the Jetty plugin has been set up in the project POM file, the jetty:run goal is available:

- mvn jetty:run

Container Configuration

- scanIntervalSeconds

The pause in seconds between sweeps of the webapp to check for changes and automatically hot redeploy if any are detected. By default this is 0, which disables hot deployment scanning. A number greater than 0 enables it.

How to do it...

```
<configuration>
  <scanIntervalSeconds>2</scanIntervalSeconds>
</configuration>
```

changeJettyPort

How to do it...

```
<configuration>
    ...
    <connectors>
      <connector implementation="org.mortbay.jetty.nio.SelectChanne
lConnector">
        <port>8088</port>
      </connector>
    </connectors>
    ...
</configuration>
```

Zuletzt geändert: Freitag, 3. Mai 2013, 18:30

NAVIGATION

Einstellungen

2521 Web Applikationen I

Maven Understanding the Project Object Model

Every Apache Maven project contains a pom.xml file. The pom.xml file is the XML representation of the project and thus contains all metadata pertaining to the project. This includes project configuration, defect tracking system details, project organization and licenses, project paths, dependencies, and so on.

The structure of a typical Apache Maven Project POM file is described as follows:

- The basics: This section contains project co-ordinates, dependency management, and inheritance details. Additionally, it also contains modules and project level properties.
- Build settings: This section contains the build details.
- Project metadata: This section contains project-specific details such as name, organization, developers, URL, inception year, and so on.
- Environment: This section contains all information regarding the environment including details of the version control being, issue management, continuous integration, mailing lists, repositories, and so on.

```
<project … >
  <modelVersion>4.0.0</modelVersion>

  <!-- The Basics -->
  <groupId>...</groupId>
  <artifactId>...</artifactId>
  <version>...</version>
  <packaging>...</packaging>
  <dependencies>...</dependencies>
  <parent>...</parent>
  <dependencyManagement>...</dependencyManagement>
  <modules>...</modules>
  <properties>...</properties>

  <!-- Build Settings -->
  <build>...</build>
  <reporting>...</reporting>

  <!-- Project Meta Data -->
  <name>...</name>
  <description>...</description>
  <url>...</url>
  <inceptionYear>...</inceptionYear>
  <licenses>...</licenses>
  <organization>...</organization>
  <developers>...</developers>
  <contributors>...</contributors>

  <!-- Environment -->
  <issueManagement>...</issueManagement>
  <ciManagement>...</ciManagement>
  <mailingLists>...</mailingLists>
  <scm>...</scm>
  <prerequisites>...</prerequisites>
  <repositories>...</repositories>
  <pluginRepositories>...</pluginRepositories>
  <distributionManagement>...</distributionManagement>
  <profiles>...</profiles>
</project>
```

Zuletzt geändert: Donnerstag, 25. April 2013, 22:08

Einstellungen

2521 Web Applikationen I

Maven Understanding the build lifecycle

The build lifecycle

explicitly defines the process of building, testing, distributing an artifact, and is at the heart of every Maven project.There are three inbuilt build lifecycles: default, clean, and site.

Default lifecycle

The default lifecycle handles the project compilation, test, and deployment. While it contains over 20 build phases, the following are the most important phases:

- Validate: validates that all project information is available and is correct
- Compile: compiles the source code
- Test: runs unit tests within a suitable framework
- Package: packages the compiled code in its distribution format
- Integration-test: processes the package in the integration-test environment
- Verify: runs checks to verify that the package is valid
- Install: installs the package in the local repository
- Deploy: installs the final package in a remote repository

Whenever you execute a build phase, all prior build phases are executed sequentially. Hence, executing mvn integration-test will execute the validate, compile, test, and package build phases before executing the integration-test build phase.

Clean lifecycle

The clean lifecycle handles the project cleaning and contains the following build phases:

- Pre-clean: executes processes required before project cleaning
- Clean: removes all files generated by previous builds
- Post-clean: executes processes required to finalize project cleaning

Site lifecycle

The site lifecycle handles the generation and deployment of the project's site documentation:

- Pre-site: executes processes required before generation of the site
- Site: generates the project's site documentation
- Post-site: executes processes required to finalize the site generation and prepares the site for deployment
- Site-deploy: deploys the site documentation to the specified web server

Zuletzt geändert: Donnerstag, 25. April 2013, 22:15

2521 Web Applikationen I

Maven and Eclipse

Eclipse integration (only for pre Kepler Version)

How to do it...

1. Go to: 'Help' -> 'Eclipse Marketplace...'
2. Find: 'maven'
3. Install 'Maven Integration for Eclipse WTP (Incubation)'

Creating a Maven project with Eclipse

How to do it...

1. 'File' -> 'New' -> 'Other...'
2. Select 'Maven' -> Maven Project
3. Select an Archetype (e.g. org.apache.maven.archetypes | maven-archetype-webapp)
4. Enter a group id for the artivact (e.g. ch.hftm.webtest)
5. Enter an artifact id (e.g. webtest)
6. go to Project Properties
7. set Java Build Path
8. set Project Facets
9. Finish

Execute Maven commands through Eclipse

How to do it...

1. selecting the Run -> Run configurations
   Select 'Maven Build' and create a new configuration(e.g. Goals: jetty:run)

Importing a Maven project with Eclipse

How to do it...

1. Launch the project import wizard by selecting File -> Import
2. Then selecting Maven -> Existing Maven Projects

Project-specific Maven options

How to do it...

1. options are available by right-clicking on the project and selecting the Maven menu item

Zuletzt geändert: Freitag, 25. April 2014, 14:09

Einstellungen

2521 Web Applikationen I

HSQLDB Installation & Konfiguration

HSQLDB (HyperSQL DataBase) is the leading SQL relational database engine written in Java. It offers a small, fast multithreaded and transactional database engine with in-memory and disk-based tables and supports embedded and server modes. It includes a powerful command line SQL tool and simple GUI query tools.

Setting up HSQLDB on Windows and Mac

How to do it...

1. download the latest version (hsqldb.org)
2. extract ./lib/hsqldb.jar to any directory

run HSQLDB Server on Windows and Mac

How to do it...

To run the server with one database (default settings)

- java -cp hsqldb.jar org.hsqldb.Server -database myDatabaseName

To run the server with more then one database (server.properties file needed)

- java -cp hsqldb.jar org.hsqldb.Server

To run the server with more then one database (server.properties file needed) extended with an additionally DB

- java -cp hsqldb.jar org.hsqldb.Server -database.10 myDB -dbname.10 myDBName

server.properties

HSQLDB is using a properties file to get the startup configuration. In general you can configure all databases you would like to use and save it to the properties file.

How to do it...

1. create a text file with the name 'server.properties'
2. save it to same directory where hsqldb.jar is located
3. add your configuration e.g.

```
server.database.0=./path/myFirstDB
server.dbname.0=myFirstDBName
server.database.1=./path/mySecondDB
server.dbname.1=mySecondDBName
server.port=9001
```

Database Manager

hsqldb provides some simple database management tools.

How to do it...

1. java -cp hsqldb.jar org.hsqldb.util.DatabaseManager (or DatabaseManagerSwing)
2. configure your connetion

- Type: HSQL Database Engine Server
- Driver: org.hsqldb.jdbcDriver
- URL: jdbc:hsqldb:hsql://localhost/ (or 'jdbc:hsqldb:hsql://localhost/myFirstDBName' if you run more than one DB)
- User: SA
- Password:

Zuletzt geändert: Freitag, 9. Mai 2014, 13:37

Maven Installation & Configuration

Setting up Apache Maven on Windows

How to do it...

1. download Maven (http://maven.apache.org)
2. extract the archive into any folder (e.g. c:\maven\)
3. add the M2_HOME environment variable (c:\maven)
4. extend the PATH variable with: %M2_HOME%\bin

Setting up Apache Maven on Mac (10.8,10.9)

How to do it...

1. download Maven (http://maven.apache.org)
2. extract the archive into any folder (e.g. /usr/local/maven/)
3. add the M2_HOME environment variable
4. sudo nano /etc/launchd.conf
5. add to launchd.conf: setenv M2_HOME /usr/local/maven
6. extend the PATH variable
7. sudo nano /etc/paths
8. add: /usr/local/maven/bin

Setting up Apache Maven on Mac (10.10)

How to do it...

1. download Maven (http://maven.apache.org)
2. extract the archive into any folder (e.g. /usr/local/maven/)
3. Create an `environment.plist` file in `~/Library/LaunchAgents/` with this content

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd'
<plist version="1.0">
<dict>
<key>Label</key>
<string>my.startup</string>
<key>ProgramArguments</key>
<array>
<string>sh</string>
<string>-c</string>
<string>

launchctl setenv JAVA_HOME /Library/Java/JavaVirtualMachines/jdk1.8.0_25.jdk/Contents/Home
</string>

</array>
<key>RunAtLoad</key>
<true/>
</dict>
</plist>

1. extend the PATH variable
2. sudo nano /etc/paths
3. add: /usr/local/maven/bin

Verifying the Apache Maven installation

How to do it...

1. mvn --version (terminal)

Zuletzt geändert: Dienstag, 5. Mai 2015, 10:04

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<persistence xmlns="http://java.sun.com/xml/ns/persistence"xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
http://java.sun.com/xml/ns/persistence/persistence_1_0.xsd"version="1.0">
  <persistence-unit name="gaestebuchDatabase" transaction-type="RESOURCE_LOCAL">
    <provider>org.hibernate.ejb.HibernatePersistence</provider>
    <properties>
      <property name="hibernate.hbm2ddl.auto" value="update"/>
      <property name="hibernate.show_sql" value="true"/>
      <property name="hibernate.cache.provider_class"value="org.hibernate.cache.HashtableCacheProvider"/>
      <property name="hibernate.dialect"value="org.hibernate.dialect.HSQLDialect"/>
      <property name="hibernate.format_sql" value="true"/>
      <property name="hibernate.use_sql_comments" value="true"/>
      <property name="hibernate.connection.driver_class"value="org.hsqldb.jdbcDriver"/>
      <property name="hibernate.connection.url"value="jdbc:hsqldb:hsql://localhost"/>
      <property name="hibernate.connection.password" value=""/>
      <property name="hibernate.connection.username" value="sa"/>
    </properties>
  </persistence-unit>
</persistence>
```